Applied Linear Models

Nick Syring

2022-11-27

Contents

Chapter 1

About

This is a collection of notes intended for students studying applied linear models at Iowa State University. Specifically, these notes are modeled after course notes ofr STAT 500, a one-semester course primarily taken by first-year statistics graduate students. This site is a work in progress.

Some relevant textbook references include Applied Linear Statistical Models, 5th ed., by Kutner, Nachtsheim, Neter, and Li, and The Statistical Sleuth, 3rd ed. by Ramsey and Schafer.

Chapter 2

Data Analysis and Statistical Inference

In this first chapter we define and discuss some important concepts regarding data and scientific investigation.

2.1 Data, Experiments, and Studies

We encounter numerical summaries of information constantly in our everyday lives and sort through these in order to make all sorts of decisions. In this section we will formalize the concept of *data* connected to scientific study. Broadly speaking, data is anything we observe that is relevant to answering a question of interest, i.e., data is tacitly assumed to be informative about the question.

Three types of studies in which data are collected and analyzed are in designed, interventional experiments, observational studies, and exploratory studies. The following illustrations help to differentiate these two types of studies. One difference we will see is all about timing—experiments start with a question of interest and then are designed to answer the question. Exploratory data analysis is often done using data collected with no particular question in mind, or some other question besides the current one. Another difference is that experiments require interventions—imposed changes of behavior or conditions on the individuals in the experiment—while observational studies do not include such interventions.

2.1.1 James Lind's Scurvy Trial

Clinical trials are familiar designed, interventional experiments. A famous, early example is James Lind's Scurvy trial. Lind was a doctor aboard a British ship.

Several sailors with him were suffering from scurvy. He selected 12 sailors in similar, poor condition, and assigned to pairs of them 6 different treatments (the intervention). The two who received oranges and lemons to eat recovered fully; those who received apple cider fared next best. Lind specifically wanted to understand which treatments would be most effective at curing scurvy and planned an experiment to answer his question. His selection of like sailors and random, paired treatment assignment constitute early examples of randomization and control in experimental design.

2.1.2 Framingham Heart Study

Named for Framingham, Massachusetts, where the participants were first recruited, this was a long-running observational study of Americans aimed at understanding risks associated with heart disease. Participants agreed to medical testing every 3-5 years, from which the study researchers concluded a number of important findings, such as cigarette smoking substantially increases the risk of heart disease. There are no interventions; the researchers simply observe the patients and make conclusions based on how the patients choose to live, e.g., tobacco use.

2.1.3 Harris Bank Sex Pay Study

93 salaries of entry-level clerical workers who started working at Harris Bank between 1969 and 1971 show men were paid more than women. (From The Statistical Sleuth, reproduced from "Harris Trust and Savings Bank: An Analysis of Employee Compensation" (1979), Report 7946,Center for Mathematical Studies in Business and Economics, University of Chicago Graduate School of Business.)

2.1.4 Large, Aggregated Data Sets

There are now many large "data sets" recording all sorts of information about consumers, e.g., data obtained by Google and other technology companies whenever consumers use their sites, apps, or devices. There are many potential use cases for such information; for instance, such data has proven helpful for targeted marketing of specific products. Such applications may be termed exploratory research—these are characterized, in part, by exploration/examination of data that was originally collected for some other purpose. In other words, the "data" was collected with either no question or some other question in mind.

2.1.5 Study Concepts

The above examples illustrate several key concepts related to scientific studies.

- Research question - There is always a reason researchers go to the trouble of collecting and analysing data; they have an important question they want to answer. For example, what can sailors do to prevent scurvy?
- Experimental units/Subjects - the research question usually references people, things, animals, or some other entity that can be studied in order to answer the question. When these are observed and measured then they are called experimental units or subjects. In the context of interventional experiments these usually refer to the units of randomization; see below.
- Data - we are inundated with information, numbers, figures, and graphs in our everyday lives. Is this data? Anything information gathered to answer a particular research question can be considered data. Relevancy to a research question is key.
- Intervention - When James Lind gave different foods to sick sailors he was making an intervention, and his goal was to study the effect of his interventions on the sailors well-being. Experiments include one or more interventions, whereas observational studies do not feature any interventions on the part of the researcher.
- Randomization - When researchers intervene, they should apply their interventions randomly with respect to subjects. In experiments the experimental units are the entities that are randomized and given interventions.
- Response/outcome variables - studies often measure multiple variables and study relationships between them. Typically the researchers expect one variable is affected by another. The response, or outcome—like the health of sailors, or pay of workers—is the variable being affected by the intervention in an experiment or by another, independent variable in an observational study.
- Control - Researchers should try to limit the effects of variables on the response that are not of interest to the study. For example, in the gender pay study, the researchers studied only entry-level workers. They *controlled* for prior experience to better isolate potential sex effects on pay.

2.1.6 Randomization, control, and causation

In experiments the researcher performs one or more interventions—such as giving patients cider versus citrus fruits in Lind's scurvy trial. The principle of *randomization* asserts that interventions in experiments should be assigned to experimental units randomly. When experimental units are heterogeneous—not all the same—it stands to reason that some of their differences apart from the intervention may impact the response to the experiment recorded by the researcher. Randomization is a way to even out these heterogeneities between groups receiving different interventions. That way, it is the intervention, rather than some other difference, which is responsible for substantially different outcomes. Randomization systematically accounts for heterogeneities, but the extent to which it works depends on the number of experimental units, the number of groups being randomized, and the presence of one or more important hetero-

geneities. For examples, consider the following: 1. Suppose in ten experimental units there is one unobserved, dichotomous trait that affects the experimental response. Three of the ten have version "0" of the trait and 7 have version "1". Randomly split the ten into two groups of five, each group to receive a different intervention. The chance all three end up in the same group is 1/6, not ignorably small... 2. On the other hand, suppose there are 100 experimental units, half have trait "0" and half trait "1". The chance at least 35 of either trait type end up in the same one half random split is ≈ 0.
Generally when an intervention is randomized over experimental units we interpret any significant difference in outcome/response between intervenion groups as having been *caused* by the intervention itself, as opposed to some other unobserved characteristic—these are sometimes called *lurking variables* or *confounding variables*. But, randomization is not foolproof; small sample sizes (few experimental units) and/or the presence of many confounding variables can reduce the effectiveness of randomization.

When the researcher knows about potential confounders ahead of time, the principle of *blocking* says experimental units should be representatively divided with respect to intervention across values of this variable. For example, if experimental units are humans both young and old, then the different interventions should be applied to equal numbers of young and old people. One way to accomplish this is to let age group be a *blocking factor*. In the case of a dichotomous intervention this means half of the old people will be randomly assigned to one intervention and half of the young people will be randomly assigned to one intervention—as opposed to randomly assigning half of all the experimental units to one intervention.

The principle of *control* states that intervention groups should be made as homogeneous as possible. When experiments are well-controlled researchers often assume that they can determine *causation*, and any observed differences in experimental outcome between intervention groups can be attributed to the intervention. Of course, as mentioned above, the ability of an experiment to determine causation is not all or nothing; rather, it depends on unknowns. Nevertheless, stronger controls make the results of experiments more trustworthy, and less likely to be caused by confounding variables.

Non-interventional, observational studies are not used to establish causative relationships. Rather, we say such studies establish *associations* between variables. For example, in the Framingham study, the researchers did not randomly assign individuals to groups of tobacco-users and non-users. Even though these days the evidence is quite strong that tobacco use causes heart disease, the bar for such a claim is much higher when the variable of interest—tobacco use—cannot be randomly assigned to experimental units. That's not to say elements of control cannot be used. For instance, if enough data is collected, it is possible to compare tobacco-users and non-users with nearly the same ages, sexes, incomes, education, living in the same zip codes, etc. The more potential confounders are explicitly controlled, the closer such an observational study comes

to a randomized experiment.

2.1.7 Populations and scope of inference

Whether conducting an experiment or collecting observational data, the units/subjects have to come from somewhere. The *population* refers to the set of all possible subjects—it may be finite or infinite, and it may be concrete or hypothetical. For examples, the population of current Iowa State undergraduates is well-defined, whereas the population of mouse kidney cells exists in a hypothetical sense. *Sampling* describes how subjects are obtained from the population for observation. *Random sampling* is any scheme involving selecting a subset of subjects from a larger group in some random fashion. A *simple random sample* of size n is obtained when every subset of n subjects from a total group is equally likely to be selected. Other types of random selection are possible, but we won't often consider these: - *stratified random sampling* obtains when simple random samples from separate groups/strata are combined, e.g., a 50/50 random sample stratified by male/female can be formed by taking a simple random sample of ten males and a simple random sample of 10 females from a group of 50 males and 50 females - *cluster random sampling* obtains when a larger group is subdivided into smaller groups and subgroups are selected at random, e.g., a cluster random sample of Iowa high schoolers can be obtained by choosing all high schoolers who attend one of a simple random sample of Iowa high schools.

Generally, conclusions about subjects in the study—whether it is an experiment or an observational study—may be assumed to hold for the wider population as a whole when the subjects are chosen randomly. The details of this *generalizability* of results depend on the type of random sampling conducted; we'll focus on the case of simple random sampling specifically. On the other hand, when subjects are not randomly sampled from the population, study results cannot be generalized back to the population. The reason is that the lack of randomness in selection implies some subsets of the population are more or less likely to be present in the sample of subjects observed, hence, that sample is not necessarily *representative* of the population. For an extreme example, consider a population of both young and old people and an experiment studying the effects of Covid-19. It's well-known Covid-19 is much more harmful to old people compared to young people. So this is a potential confounder. If we select only young people to study, then we certainly cannot claim the results would be similar had we studied both young and old people.

Non-random sampling schemes are quite common because they are usually easier and cheaper to implement than random sampling schemes. A *convenience sample* is just what it sounds like—a rule that selects subjects that are easy to select—such as conducting a poll of your closest friends. When a non-random sample is used, remember that the results cannot be interpreted beyond the group subjects that were observed.

Sometimes a researcher intends to study one population, but obtains data from another population. This mismatch is important to identify as it can cause bias—which simply means the answer to the researcher's question is different for the population for which data is observed compared to the intended population. As in the extreme example above, effects of Covid-19 are different in old and young populations, so the results from an experiment studying only the young are biased when viewed from the perspective of the population of old and young combined.

2.2 Data Summaries

Data may take on many forms including sound waves, images comprised of pixels/voxels, and graphs of functions/surfaces. We will almost exclusively consider data that may be represented in a tabulated format, like the following data on weights of chickens.

```
##   weight Time Chick Diet
## 1     42    0     1    1
## 2     51    2     1    1
## 3     59    4     1    1
## 4     64    6     1    1
## 5     76    8     1    1
## 6     93   10     1    1
```

Regardless of the type of data, few users can make sense of all the data at once; rather, we need data summaries—numbers or plots that point out important features of the whole data set.

2.2.1 Numerical Summaries

We briefly describe summaries for categorical and numerical, continuous data.

Suppose a variable X takes categorical values, e.g., $X \in \{0, 1, 2, 3\}$. A data set consisting of observed x values, may be summarized by tabulating proportions, i.e., of 100 observations, 0.41 were $x = 0$, 0.29 were $x = 1$, 0.17 were $x = 2$ and 0.13 were $x = 3$. Binary or dichotomous variables may be also be summarized by odds and odds ratios. Suppose $Y \in \{0, 1\}$. The observations $y_1, \ldots, y_n$ may be summarized by saying 60% were $y = 1$ versus 40% $y = 0$ or, equivalently, the observed odds of $y = 1$ was $0.6/0.4 = 1.5$. Suppose response Y is *blocked* by another dichotomous variable, say, undergraduate versus graduate status, and suppose the observed odds for undergraduates is 1.5 while the observed odds for graduates is 0.8. Then, the observed odds ratio for undergraduates versus graduates is $1.5/0.8 = 1.875$.

In contrast to categorical data, continuous data takes values in the real numbers $\mathbb{R}$ or some interval of real numbers. Most numerical summaries of continuous variables either measure location or dispersion. Location refers, in one way or another, to a typical or average value while dispersion refers to the spread of the values. Common measures of location are the mean, median, and trimmed mean:

- The mean of $x_1, \ldots, x_n$ is $\overline{x} = n^{-1} \sum_{i=1}^{n} x_i$.
- The median is denoted $\tilde{x}$. Sort the x's from least to greatest labeling them as $x_{(1)}, \ldots, x_{(n)}$. Then, $\tilde{x} = x_{(\frac{n+1}{2})}$ if n is odd and $\tilde{x} = \frac{1}{2}\left(x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)}\right)$ if n is even.
- The $\alpha\%$ trimmed mean is the mean of the remaining x values after ignoring the smallest and largest $\alpha\%$ of values. Roughly, the $\alpha\%$ trimmed mean averages $x_{(1+\alpha n)}, \ldots, x_{n-\alpha n-1}$.
- Besides the median, we may be interested in "locations" besides the center, and these could be summarized using quantiles. Observed quantiles may be defined in a number of ways. One definition says the $\alpha \in (0, 1)$ quantile is the smallest observed x value such that at least $\alpha\%$ of observed x values are no more than x.

Common measures of dispersion include the variance and standard deviation, the range, and the interquartile range:

- The observed variance of $x_1, \ldots, x_n$ is given by $\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2$ and may be equivalently computed as $\frac{1}{n-1}\left\{\sum_{i=1}^{n} (x_i^2) - n\overline{x}^2\right\}$.

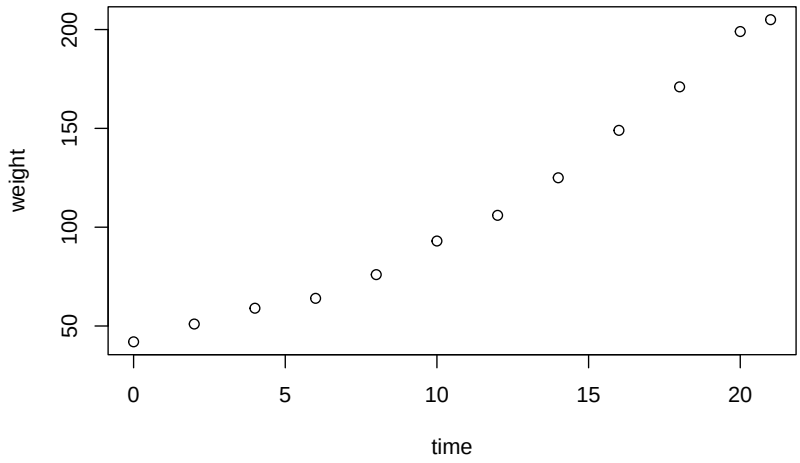
- In the calculation of variance the x values are squared, which means the variance has units equal to the squared units of x; that is, if x is measured in millimeters then its variance is measured in millimeters squared, or, e.g., dollars then dollars squared. Often the squared units are difficult to interpret. Instead, we define the standard deviation as the square root of the variance, which shares the units of the observed x variable.

- The range is the difference between maximum and minimum values, i.e., $x_{(n)} - x_{(1)}$.
- The interquartile range (IQR) is the difference between the 0.75 and 0.25 quantiles.

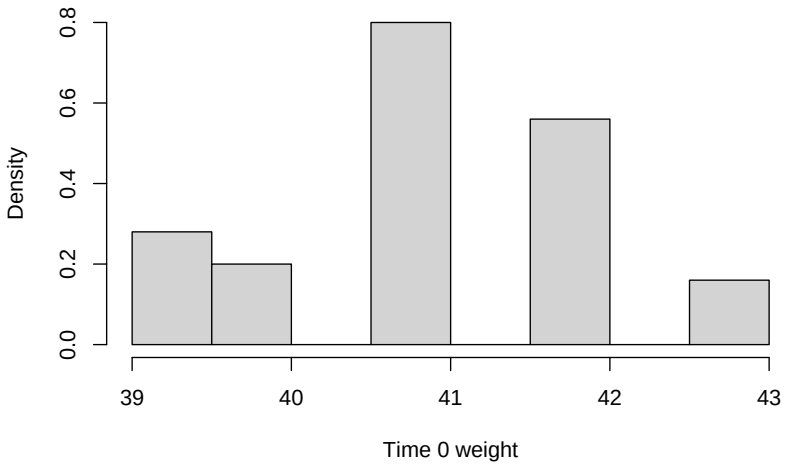
2.2.2 Visual Summaries

Plots and graphs can also be used to summarize a data set or particular observed variable. Three plots we will use extensively in the course are the scatterplot, histogram, and quantile-quantile (qq) plot.

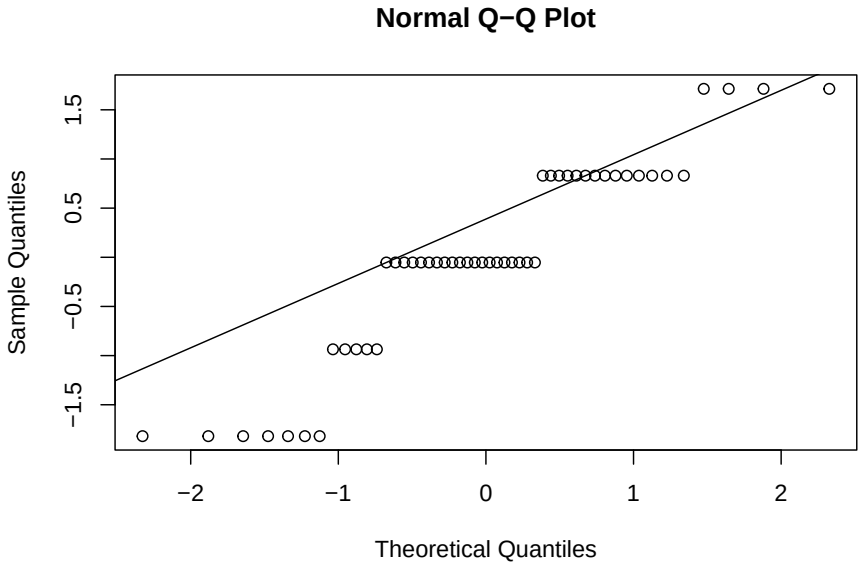
- For paired observations of two variables $(x_i, y_i),\ i = 1, \ldots, n$ a scatterplot displays (x_i, y_i) in the xy-plane. For example, see the plot of a chicken's weight versus time from the tabulated data set above. Scatterplots are useful for assessing relationships between variables. For example, the chick's weight increases with time, maybe in a linear or slightly quadratic fashion.



- For a single variable a histogram summarizes the distribution of its observed values by counting the numer of observations in different intervals (buckets) of values. Keep in mind that histograms with different choices of buckets may look very different. Check out this histogram of "Time 0" weights of all 23 chicks.



- A qq-plot compares the shape of a distribution of observed values to another known distribution, often the standard normal distribution. For example, make the standardizing transformation $(x_i - \overline{x})/\hat{\sigma}_x$ where x_i is the Time 0 weight of chick i, $\overline{x}$ is the observed mean and $\hat{\sigma}_x$ is the observed standard deviation of those values. Compute the α quantile of these values or several α values in $(0, 1)$ along with the corresponding standard normal quantiles (z-scores). Plot the pairs of α quantiles in the xy-plane. If the standardized weights are approximately normal, then the points should lie approximately on the line $y = x$. Note that extreme quantiles are always less reliably estimated, so it is typical for the ends of the "line" to fray up or down from the diagonal.



2.3 Statistical Inference

Summarizing data sets is important because no one can make sense of more than a few numbers at a time. But, data summaries cannot by themselves answer our research questions. That is because data summaries only say something about the particular data set we observe, while our research questions concern the whole population. Recall, a major aim of experimentation is generalizing from observations to population. When we make such generalizations we often refer to them as *inferences*; and, statistical inferences are characterized by their careful treatment of statistical concepts, such as hypotheses, and Type 1 and 2 errors, which we discuss below.

A hypothesis is a claim or assertion about a population. These may be simple, i.e., "the population mean is 5", or more complex, like "the population distribution of values is equivalent to a Normal probability distribution". Notice that both of these statements are either true or false, yes or no. A hypothesis then may be called the "null hypothesis", and its complement (opposite) the alternative hypothesis. Which is which depends on the context. We make educated guesses about the truthfulness of a null hypothesis based on observations/data. Our educated guesses may be right or wrong, but we will never know because we will never "see" the whole population. If we reject the null hypothesis as false when it really is true, then we make a Type 1 error. The opposite, keeping the null hypothesis in favor over its alternative, when it is actually false, is a Type 2 error. Ideally, we would make no errors, but that's not possible. In fact,

the two errors have an inverse relation. For example, if we adopt the rule that we always reject the null hypothesis, then we will necessarily maximize Type 1 errors but have no Type 2 errors. And, if we take the opposite approach, then we maximize Type 2 errors while making no Type 1 errors.

Much of this course will focus on constructing tests of relevant hypotheses with the property that we limit the chance of making a Type 1 error. By chance we refer to the probability distribution of the test outcome induced by random sampling of data from the population. A test that has chance no more than α of making a Type 1 error is called a "level α test of H_0", the null hypothesis.

Chapter 3

Introduction to Linear Models

In this section we define linear models, provide simple examples, and analyze linear models for one- and two-sample problems.

3.1 Defining the linear model

Every linear model defines a linear relationship between an independent variable Y and a dependent variable X, including a random term ϵ:

$$Y = X\beta + \epsilon \tag{3.1}$$

Usually, X is a fixed or non-random variable, while ϵ is a random variable representing variation due to a random sampling mechanism, so that Y is a random outcome. Further, in (3.1) $Y = (Y_1, \dots, Y_n)^\top$ is an $n \times 1$ vector of outcomes/responses, X is an $n \times p$ matrix of fixed variables/covariates ($p < n$), $\epsilon = (\epsilon_1, \dots, \epsilon_n)^\top$ is an $n \times 1$ vector of random variables, and $\beta = (\beta_1, \dots, \beta_p)^\top$ is a $p \times 1$ coefficient vector of unknown parameters.

The *least-squares model* or just called the *linear model* is the above model with few or no additional assumptions although, to estimate the unknown parameter β, which characterizes the relationship between X and Y, assuming $E(\epsilon_i) = 0$ is very helpful and usually reasonable.

The *Gauss-Markov model*—which we will tacitly use throughout the course and examine in detail at the end of the semester—makes the assumptions $E(\epsilon_i) = 0$, $E(\epsilon_i^2) = \sigma^2$, which means the "error" term ϵ has the same variance for each random sample (homogeneous or constant variance), and $E(\epsilon_i \epsilon_j) = 0$. Or, in other words, the last two assumptions may be written $Cov(\epsilon) = \sigma^2 I_n$ where I_n is the $n \times n$ identity matrix.

3.2 Gauss-Markov model for one sample

Let P be a normal population with mean β and variance σ^2. Let $Y_i \overset{iid}{\sim} P$. Then, we may write

$$\begin{aligned} Y_i &= \beta x_i + \epsilon_i, \quad i = 1, \ldots, n, \text{ or} \\ Y &= X\beta + \epsilon \end{aligned} \tag{3.2}$$

where $x_i = 1$, so that $X = (1, 1, ..., 1)^\top$ is an $n \times 1$ vector of ones, and where $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$. This shows the Gauss-Markov model contains the one-sample normal population inference problem.

For inference on the population mean β we typically consider the point estimator $\hat{\beta} = \overline{Y}$, the sample mean. And, we evaluate the usual one-sample Student's t test of $H_0 : \beta = \beta_0$ vs. $H_a : \beta \neq \beta_0$ by comparing the test statistic

$$T = \frac{\overline{Y} - \beta_0}{\sqrt{S^2/n}}$$

to quantiles of a Student's t distribution with $n - 1$ df, where $S^2 = (n - 1)^{-1} \sum_{i=1}^{n} (Y_i - \overline{Y})^2$.

A common theme in the study and use of linear models is "model-checking". The idea is to us the data to interrogate the Gauss-Markov assumptions: linearity, normality, and constant variance (and sometimes independence). In the one-sample problem linearity is of no concern because x is constant, so there's no relationship between X and Y to check for linearity. Normality may be checked by constructing a qq-plot of the residuals $\hat{\epsilon}_i = Y_i - \hat{Y}_i$ where $\hat{Y}_i = \hat{\beta} x_i$; in this case, $\hat{\epsilon}_i = Y_i - \overline{Y}$. Often times data exhibit non-homogeneous variance in the sense that the variance increases or decreases in X. Again, this kind of non-homogeneity is not relevant in the one-sample problem. Finally, in data sets with time or space variation (like gasoline prices in Ames from 2021-2022), we may suspect non-independence of ϵ_i's. This kind of time-series dependence may be checked using a serial correlation plot; see below.

Example: A horticulturist plants 500 seeds of *centaurea cyanus* in a 10 foot by 1 foot plot of clay-type soil in 1 foot by one foot squares. After 6 weeks she records the number of sprouted plants in each square, finding

37 24 31 21 9 21 15 37 31 26.

Assuming a normal population, she wants to infer the true average number of sprouted (germinated) seeds per one-foot by one-foot plot of fifty in clay-type soil.

In particular, she'd like to know if the true mean is at least 25. She will not make a wholesale purchase of this particular seed unless she is convinced the true mean is at least 25.

The horticulturist's Gauss-Markov model has Y_i equal to the recorded number of sprouts for sub-plots $i = 1, \ldots, 10$, and where $x_i = 1$ for each. The following t-test concludes (at level 0.05) that the true mean is quite plausibly less than 25.

```
y <- c(37, 24, 31, 21,  9, 21, 15, 37, 31, 26)
ybar <- mean(y)
ybar
```

```
## [1] 25.2
```

```
s2 <- var(y)
s2
```

```
## [1] 83.28889
```

```
n <- length(y)
T <- (ybar - 25)/sqrt(s2/n)
T
```

```
## [1] 0.06930051
```

```
qt(0.95,n-1)
```

```
## [1] 1.833113
```

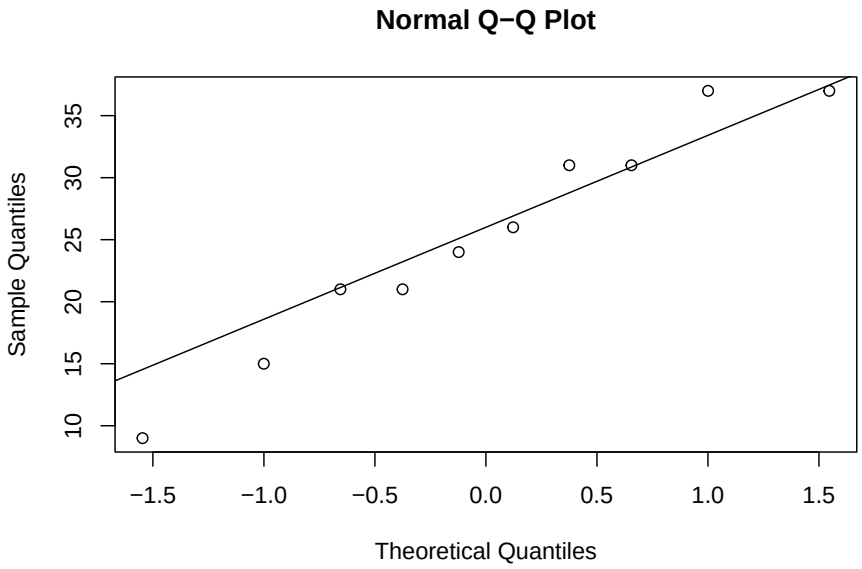
```
1-pt(T,n-1)
```

```
## [1] 0.4731329
```

```
ybar+qt(c(0.025,0.975),n-1)*sqrt(s2/n)
```

```
## [1] 18.67146 31.72854
```

Further, we can interrogate the normality (Gaussianity) assumption using a qq-plot. The plot reveals no concerns about the normality assumption.



In this particular example, we may be concerned that sub-plots next to each other may have (spatially) correlated responses. One way to evaluate this is to compute the correlation between adjacent pairs. The observed (sample) correlation is only about 12%, not high enough to suspect spatial non-independence.

```
y <- c(37, 24, 31, 21,  9, 21, 15, 37, 31, 26)
n <- length(y)
y_adj <- c(24, 31, 21,  9, 21, 15, 37, 31, 26)
cor(y[1:(n-1)], y_adj)
```

```
## [1] 0.1275145
```

3.3 Gauss-Markov model for comparing two samples

Now suppose we have two normal populations, with the same variances, and we want to compare possibly different means. For some usual notation we might say $X_{1,i} \overset{iid}{\sim} N(\mu_1, \sigma^2)$ for $i = 1, \ldots, n_1$ and $X_{2,j} \overset{iid}{\sim} N(\mu_2, \sigma^2)$ for $j = 1, \ldots, n_2$. There are (at least) two ways to represent this data in linear model format. Either way, we let $Y = (X_{1,1}, \ldots, X_{1,n_1}, X_{2,1}, \ldots, X_{2,n_2})^\top$ be the vector of $n = n_1 + n_2$ responses. The *means model* uses design matrix $X = (X_1, X_2)$ where X_1 and X_2 are $n \times 1$ column vectors with $X_1 = (1_{n_1}^\top, 0_{n_2}^\top)^\top$ and $X_2 = (0_{n_1}^\top, 1_{n_2}^\top)^\top$. Suppose $n_1 = n_2 = 10$; then, X looks like the following:

```
X = cbind(c(rep(1,10),rep(0,10)), c(rep(0,10),rep(1,10)))
X
```

```
##       [,1] [,2]
##  [1,]    1    0
##  [2,]    1    0
##  [3,]    1    0
##  [4,]    1    0
##  [5,]    1    0
##  [6,]    1    0
##  [7,]    1    0
##  [8,]    1    0
##  [9,]    1    0
## [10,]    1    0
## [11,]    0    1
## [12,]    0    1
## [13,]    0    1
## [14,]    0    1
## [15,]    0    1
## [16,]    0    1
## [17,]    0    1
## [18,]    0    1
## [19,]    0    1
## [20,]    0    1
```

The product $X\beta$ is the vector $(\beta_1 1_{n_1}^\top, \beta_2 1_{n_2}^\top)^\top$. Taking the expectation of $X\beta+\epsilon$ we get $E(Y) = (\beta_1 1_{n_1}^\top, \beta_2 1_{n_2}^\top)^\top$. Therefore, $(\beta_1, \beta_2) = (\mu_1, \mu_2)$. In other words, the linear model coefficient vector is identical to the group means.

The second way to formulate the linear model is to encode the design matrix using the *effects model*. Let $X = (1_n^\top, (1_{n_1}^\top, 0_{n_2}^\top))^\top$, or, in other words:

```
X = cbind(rep(1,20), c(rep(1,10),rep(0,10)))
X
```

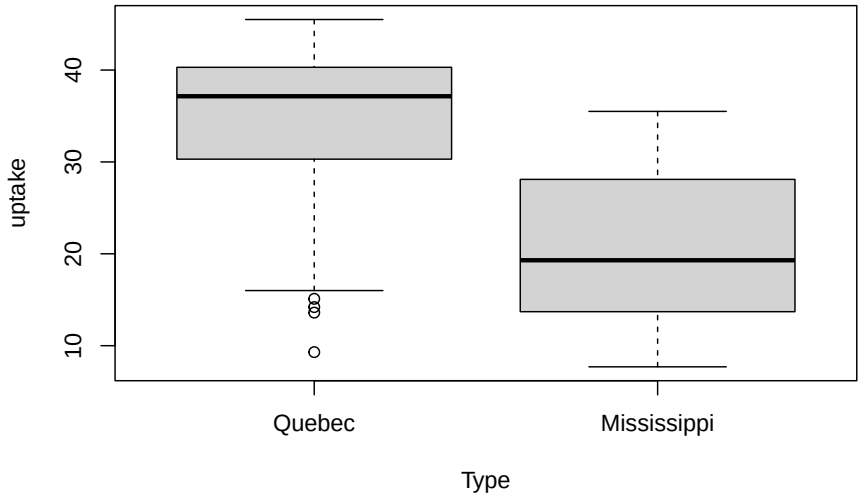
```
##       [,1] [,2]
##  [1,]    1    1
##  [2,]    1    1
##  [3,]    1    1
##  [4,]    1    1
##  [5,]    1    1
##  [6,]    1    1
##  [7,]    1    1
##  [8,]    1    1
##  [9,]    1    1
```

```
## [10,]    1    1
## [11,]    1    0
## [12,]    1    0
## [13,]    1    0
## [14,]    1    0
## [15,]    1    0
## [16,]    1    0
## [17,]    1    0
## [18,]    1    0
## [19,]    1    0
## [20,]    1    0
```

In that case, $E(Y) = E(X\beta+\epsilon) = ((\beta_1+\beta_2)1_{n_1}^\top, \beta_1 1_{n_2}^\top)^\top$. That means $\beta_1+\beta_2 = \mu_1$ and $\beta_1 = \mu_2$. Or, equivalently, $\beta_1 = \mu_2$ and $\beta_2 = \mu_1 - \mu_2$. For this design, testing $\beta_2 = 0$ is equivalent to testing for no difference in population means.

Example: The following is an analysis of carbon dioxide uptake rates of Echinochloa crus-galli grown in Quebec and Mississippi.

```
boxplot(uptake~Type, data = CO2)
```




```
# means model
hat.beta1 <- mean(CO2$uptake[CO2$Type == 'Quebec'])
hat.beta1
```

```
## [1] 33.54286
```

```
hat.beta2 <- mean(CO2$uptake[CO2$Type == 'Mississippi'])
hat.beta2
```

```
## [1] 20.88333
```

```
S2 <- (var(CO2$uptake[CO2$Type == 'Quebec']) + var(CO2$uptake[CO2$Type == 'Mississippi'])) / 2
S2
```

```
## [1] 77.33465
```

```
n <- length(CO2$uptake)
t = (hat.beta1 - hat.beta2)/sqrt(S2*(1/(n/2) + 1/(n/2)))
t
```

```
## [1] 6.596901
```

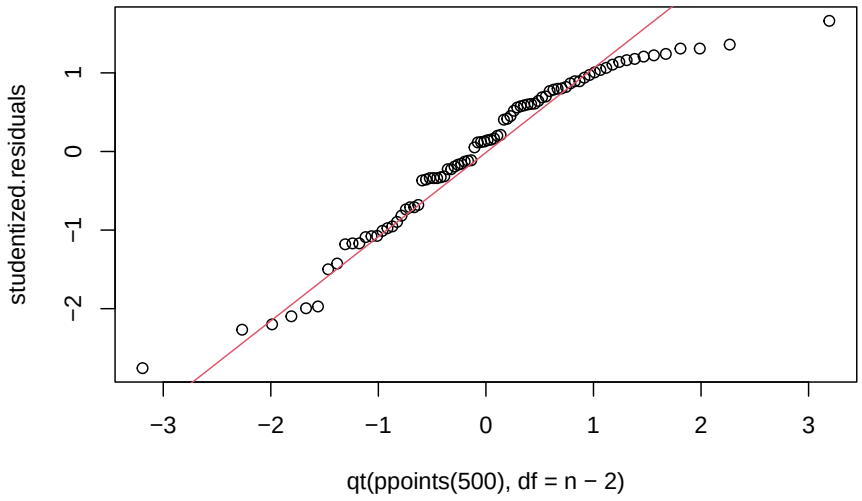
```
1-pt(t, n-2)
```

```
## [1] 1.917343e-09
```

```
qt(c(0.025,0.975),n-2)
```

```
## [1] -1.989319  1.989319
```

```
residuals <- c((CO2$uptake[CO2$Type == 'Quebec'] - hat.beta1), CO2$uptake[CO2$Type == 'Mississipp
studentized.residuals <- residuals / sqrt(S2)
qqplot(x = qt(ppoints(500), df = n-2), studentized.residuals)
qqline(studentized.residuals, distribution = function(p) qt(p, df = n-2), probs = c(0.2, 0.8), co
```



We conduct a two-sample t-test (assuming equal variances) and conclude there is a substantial difference in CO2 uptake of this grass species between the two locations. Inspection of the qq-plot reveals no concerns about normality.

```
# Effects Model
summary(lm(uptake~Type, data = CO2))
```

```
##
## Call:
## lm(formula = uptake ~ Type, data = CO2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.243  -6.243   1.187   7.027  14.617
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       33.543      1.357  24.719  < 2e-16 ***
## TypeMississippi  -12.660      1.919  -6.597 3.83e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.794 on 82 degrees of freedom
## Multiple R-squared:  0.3467, Adjusted R-squared:  0.3387
## F-statistic: 43.52 on 1 and 82 DF,  p-value: 3.835e-09
```

```
M <- lm(uptake~Type, data = CO2)
model.matrix(M)
```

```
##    (Intercept) TypeMississippi
## 1            1                0
## 2            1                0
## 3            1                0
## 4            1                0
## 5            1                0
## 6            1                0
## 7            1                0
## 8            1                0
## 9            1                0
## 10           1                0
## 11           1                0
## 12           1                0
## 13           1                0
## 14           1                0
## 15           1                0
## 16           1                0
## 17           1                0
## 18           1                0
## 19           1                0
## 20           1                0
## 21           1                0
## 22           1                0
## 23           1                0
## 24           1                0
## 25           1                0
## 26           1                0
## 27           1                0
## 28           1                0
## 29           1                0
## 30           1                0
## 31           1                0
## 32           1                0
## 33           1                0
## 34           1                0
## 35           1                0
## 36           1                0
## 37           1                0
## 38           1                0
## 39           1                0
## 40           1                0
## 41           1                0
```

```
## 42           1                 0
## 43           1                 1
## 44           1                 1
## 45           1                 1
## 46           1                 1
## 47           1                 1
## 48           1                 1
## 49           1                 1
## 50           1                 1
## 51           1                 1
## 52           1                 1
## 53           1                 1
## 54           1                 1
## 55           1                 1
## 56           1                 1
## 57           1                 1
## 58           1                 1
## 59           1                 1
## 60           1                 1
## 61           1                 1
## 62           1                 1
## 63           1                 1
## 64           1                 1
## 65           1                 1
## 66           1                 1
## 67           1                 1
## 68           1                 1
## 69           1                 1
## 70           1                 1
## 71           1                 1
## 72           1                 1
## 73           1                 1
## 74           1                 1
## 75           1                 1
## 76           1                 1
## 77           1                 1
## 78           1                 1
## 79           1                 1
## 80           1                 1
## 81           1                 1
## 82           1                 1
## 83           1                 1
## 84           1                 1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
```

```
## attr(,"contrasts")$Type
## [1] "contr.treatment"
```

The **lm** function in R fits linear models and tests $\beta_j = 0$ for each $j = 1, \ldots, p$. Note that the default design is a form of effects model, where the first level of the categorical variable is the baseline or reference level, so that the intercept term β_1 is identified with its mean, while the other β's become *effects*, the differences $\mu_2 - \mu_1$ and so on…

3.4 Pairwise testing and Bonferroni correction

The two-sample t-test is appropriate for comparing two populations; for example, when experimenters compare two interventions/treatments in terms of mean effects. But, the two-sample t-test procedure does not generalize well to comparing 3 or more populations, i.e., more interventions. Let's see why not, and then consider what we could do instead.

Consider k treatments corresponding to k populations. Two sample t-tests may be used with the null hypothesis $H_0 : \mu_i = \mu_j$, $i, j = 1, \ldots, k$, $i \neq j$. There are $\binom{k}{2}$ pairs of treatments, so there are the same number of tests to conduct. Now, suppose all the means are equal, i.e. the treatments are indistinguishiable on the basis of mean. If each test is conducted at level α then the chance of at least 1 type 1 error occurring is approximately $1 - (1 - \alpha)^{\binom{k}{2}}$, that is, if all tests are independent. For large k, this *familywise* type 1 error (type 1 error for all the tests) gets close to 1! In other words, the strategy of evaluating the hypothesis $H_0 :$ all μ_i's are equal by pairwise testing is not effective.

Several modifications are available to deal with this *multiple testing problem*. We'll discuss two: Bonferroni and Benjamini-Hochberg adjustments. The Bonferroni adjustment requires the Type-1 error level α of each test be changed to $\alpha^\star = \alpha/k$ where k is the total number of tests. By the same probability calculation (assuming independence of tests) it follows that the family-wise type 1 error probability using the Bonferroni procedure is about α:

$$\begin{aligned} P(\text{at least one rejection given } H_0 \text{ true}) &= 1 - P(\text{no rejections given } H_0 \text{ true}) \\ &\overset{ind.}{=} 1 - (1 - \frac{\alpha}{k})^k \\ &\approx 1 - e^{-\alpha} \quad \text{for large k} \\ &\approx 1 - (1 - \alpha) = \alpha \quad \text{for small } \alpha. \end{aligned}$$

The downside to Bonferroni is that when k is large each test has a very, very small chance of rejecting the null. In other words, each test has very low power. This inspired other procedures, like Benjamini-Hochberg (BH). Rather than controlling the family-wise type 1 error rate the BH procedure makes a compromise, and seeks to limit something called the false-discovery rate (FDR).

Let V be the number of tests out of k that reject the null incorrectly, and let R be the number of tests out of k that reject the null. Then, FDR is defined as $E(\frac{V}{R}|R > 0)P(R > 0)$. In words, the FDR is controlling the proportion of rejections that are incorrect out of all rejections, rather than out of all tests (like FWER does). If we did 1000 tests and rejected 40 nulls and controlled FDR at 5% we would expect 2 of the 40 rejections to be false rejections. To control FDR, the BH procedure instructs us to: 1. Sort the realized p-values from least to greatest $p_{(1)}, p_{(2)}, ..., p_{(k)}$ 2. Find the largest ℓ such that $p_{(\ell)} \leq \frac{\ell}{k}\alpha$. 3. Reject all nulls corresponding to p-values $p_{(1)}$ to $p_{(\ell)}$.

3.4.1 Example: Co2 uptake in Echinochloa crus-galli

Consider the four interventions on the echinochloa grass formed by crossing the factors location and chilling with levels Mississippi/Quebec and chilled/non-chilled. We must make 6 pairwise comparisons, which we do below, using uncorrected, Bonferroni-corrected, and BH-corrected two-sample t-tests (assuming equal variances), and based on $\alpha = 0.05$.

```
library(datasets)
xbar1 <- mean(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == "chilled"])
xbar2 <- mean(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == "nonchilled"])
xbar3 <- mean(CO2$uptake[CO2$Type=='Mississippi' & CO2$Treatment == "chilled"])
xbar4 <- mean(CO2$uptake[CO2$Type=='Mississippi' & CO2$Treatment == "nonchilled"])
t1 <- (xbar1 - xbar2)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == '
t2 <- (xbar1 - xbar3)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == '
t3 <- (xbar1 - xbar4)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == '
t4 <- (xbar2 - xbar3)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == '
t5 <- (xbar2 - xbar4)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Quebec' & CO2$Treatment == '
t6 <- (xbar3 - xbar4)/sqrt((0.5*(var(CO2$uptake[CO2$Type=='Mississippi' & CO2$Treatment
p1 <- 2*(1-pt(abs(t1),40))
p2 <- 2*(1-pt(abs(t2),40))
p3 <- 2*(1-pt(abs(t3),40))
p4 <- 2*(1-pt(abs(t4),40))
p5 <- 2*(1-pt(abs(t5),40))
p6 <- 2*(1-pt(abs(t6),40))
p1
```

```
## [1] 0.2348577
```

```
p2
```

```
## [1] 2.000916e-08
```

```
p3
```

```
## [1] 0.03471652
```

```
p4
```

```
## [1] 1.297742e-10
```

```
p5
```

```
## [1] 0.001011226
```

```
p6
```

```
## [1] 2.355328e-06
```

```
sorted <- cbind(1:6, round(c(p1,p2,p3,p4,p5,p6), 4))
sorted <- sorted[order(sorted[,2]),]
cbind(sorted,c(0.05/6,0.05*2/6,0.05*3/6,0.05*4/6,0.05*5/6,0.05*6/6), c(sorted[1,2] <= 0.05*1/6, s
      c(sorted[1,2] <= 0.05/6, sorted[2,2] <= 0.05/6, sorted[3,2] <= 0.05/6, sorted[4,2] <= 0.05/
```

```
##      [,1]   [,2]        [,3] [,4]        [,5] [,6]
## [1,]    2 0.0000 0.008333333    1 0.008333333    1
## [2,]    4 0.0000 0.016666667    1 0.008333333    1
## [3,]    6 0.0000 0.025000000    1 0.008333333    1
## [4,]    5 0.0010 0.033333333    1 0.008333333    1
## [5,]    3 0.0347 0.041666667    1 0.008333333    0
## [6,]    1 0.2349 0.050000000    0 0.008333333    0
```

Using uncorrected tests we would reject the null hypothesis in all but the first test. We actually get the same result using the BH procedure. but, the Bonferroni procedure is more conservative and does not allow us to reject the null in the 1st or 3rd tests.

3.5 Linear Models for more than two treatments (populations)

We may generalize the "means model" and "effects model" constructions to apply the Gauss-Markov model to data sets with several treatments, as in the CO2

data set considered above. Recall that the CO2 data set has 84 observations, 21 in each of four groups: Quebec-chilled, Quebec-nonchilled, Mississippi-chilled, and Mississippi-nonchilled. If we sort the response vector to correspond to these categories, we can construct the design matrix as follows: $X = (X_1, X_2, X_3, X_4)$ where X_1 is the 84×1 column vector combined by stacking a 21×1 vector of 1's atop a 63×1 vector of 0's. The remaining columns of X are formed analogously, with the 21 1 values occupying the 22nd through 42nd spots, the 43rd through 63rd, and the 64th through 84th, respectively.

```
X <- cbind(  c(rep(1,21),rep(0,21),rep(0,21),rep(0,21)),
             c(rep(0,21),rep(1,21),rep(0,21),rep(0,21)),
             c(rep(0,21),rep(0,21),rep(1,21),rep(0,21)),
             c(rep(0,21),rep(0,21),rep(0,21),rep(1,21)))
```

This construction is a "means model" matrix because the coefficient vector β is identified with the four group means. That is, $X\beta = (\beta_1 1_{21\times1}^\top, \beta_2 1_{21\times1}^\top, \beta_3 1_{21\times1}^\top, \beta_4 1_{21\times1}^\top)^\top$. So, $\beta_j = \mu_j$ for $j = 1, ..., 4$.

There are many ways to define an "effects model" for the CO2 dataset. Recall that R uses an effects model parametrization by default. Let's fit a linear model in R and see what model matrix R constructs.

```
library(datasets)
CO2subset <- data.frame(CO2$Type, CO2$Treatment,  CO2$uptake)
my.lm <- lm(CO2.uptake~CO2.Type+CO2.Treatment, data = CO2subset)
model.matrix(my.lm)
```

```
##    (Intercept) CO2.TypeMississippi CO2.Treatmentchilled
## 1            1                   0                    0
## 2            1                   0                    0
## 3            1                   0                    0
## 4            1                   0                    0
## 5            1                   0                    0
## 6            1                   0                    0
## 7            1                   0                    0
## 8            1                   0                    0
## 9            1                   0                    0
## 10           1                   0                    0
## 11           1                   0                    0
## 12           1                   0                    0
## 13           1                   0                    0
## 14           1                   0                    0
## 15           1                   0                    0
## 16           1                   0                    0
## 17           1                   0                    0
```

```
## 18          1          0          0
## 19          1          0          0
## 20          1          0          0
## 21          1          0          0
## 22          1          0          1
## 23          1          0          1
## 24          1          0          1
## 25          1          0          1
## 26          1          0          1
## 27          1          0          1
## 28          1          0          1
## 29          1          0          1
## 30          1          0          1
## 31          1          0          1
## 32          1          0          1
## 33          1          0          1
## 34          1          0          1
## 35          1          0          1
## 36          1          0          1
## 37          1          0          1
## 38          1          0          1
## 39          1          0          1
## 40          1          0          1
## 41          1          0          1
## 42          1          0          1
## 43          1          1          0
## 44          1          1          0
## 45          1          1          0
## 46          1          1          0
## 47          1          1          0
## 48          1          1          0
## 49          1          1          0
## 50          1          1          0
## 51          1          1          0
## 52          1          1          0
## 53          1          1          0
## 54          1          1          0
## 55          1          1          0
## 56          1          1          0
## 57          1          1          0
## 58          1          1          0
## 59          1          1          0
## 60          1          1          0
## 61          1          1          0
## 62          1          1          0
## 63          1          1          0
```

```
## 64           1                    1                    1
## 65           1                    1                    1
## 66           1                    1                    1
## 67           1                    1                    1
## 68           1                    1                    1
## 69           1                    1                    1
## 70           1                    1                    1
## 71           1                    1                    1
## 72           1                    1                    1
## 73           1                    1                    1
## 74           1                    1                    1
## 75           1                    1                    1
## 76           1                    1                    1
## 77           1                    1                    1
## 78           1                    1                    1
## 79           1                    1                    1
## 80           1                    1                    1
## 81           1                    1                    1
## 82           1                    1                    1
## 83           1                    1                    1
## 84           1                    1                    1
## attr(,"assign")
## [1] 0 1 2
## attr(,"contrasts")
## attr(,"contrasts")$CO2.Type
## [1] "contr.treatment"
##
## attr(,"contrasts")$CO2.Treatment
## [1] "contr.treatment"
```

Based on the model matrix, we see that R defines three coefficients, $\beta_1, \beta_2, \beta_3$.

Chapter 4

Randomization and Permutation Testing

4.1 Setup

Randomization/permutation testing is performed in the context of randomized intervention experiments or observational studies to examine the differences in response between two groups. The setting is similar to the setting in which the two-sample t-test is often used, but there are few other similarities.
A general algorithm for a randomization test of two groups, labelled, say, 0 and 1, is as follows: 1. Choose some statistic, say T, quantifying the response difference between the groups and compute this $T = t_0$ for the observed data. For m in 1:M repeat: 2. Permute the 0,1 labels of the responses. If the data comes from a randomized experiment this is like saying the randomization of subjects to treatments turned out differently, hence the name randomization test. 3. Compute the m^{th} statistic value t_m on the responses with permuted/re-randomized labels.

At the end of this algorithm we end up with the values $t_1, \dots, t_m$. What do we make of these? Here's the idea. For a concrete example, suppose T is the Student's t two-sample test statistic, i.e., $T = \frac{\overline{x}_0 - \overline{x}_1}{\sqrt{S_0^2/n_0 + S_1^2/n_1}}$. Let x denote all the responses of both groups pooled together, i.e., $x = (x_{0,1}, ..., x_{0,n_0}, x_{1,1}, ..., x_{1,n_1})$. x has some sample mean value; call it $\overline{x}$. If we randomly label n_0 of the x's 0 and the other n_1 of them 1, we should expect the average value of $\overline{x}_0 - \overline{x}_1$ to be about 0 when averaging over repeated re-labeling. That's because $\overline{x}_0$ and $\overline{x}_1$ are both approximately unbiased for $\overline{x}$. So the distribution/histogram of the $t_1, ..., t_m$ values will look roughly symmetric around zero. Then, we compare the observed value t_0 to this distribution and observe whether t_0 is a "typical" value or and "extreme" value compared to $t_1, ..., t_m$; that is, we compare t_0 the

the quantiles of $t_1, ..., t_m$. If t_0 is extreme compared to these values, then either we observed such a value by a small chance or the labels really do have an effect on the responses which is reflected in the statistic T.

This description should sound very similar to null hypothesis testing and rejection rules defined by comparing a test statistic to a null distribution. But,there are some differences. In this case, the "population" consists only of the observed values. And, the *randomization distribution*—the histogram/quantiles of $t_1, ..., t_m$—is only related to the observed data, not any larger population. This means that the "null hypothesis" that a randomization test tests is related only to the observed data. I think of the null and alternative hypotheses as the following: H_0 : the observed value t_0 is consistent with labels being randomly assigned to responses versus H_a : the observed value t_0 is NOT consistent with labels being randomly assigned to responses. Because the test references no population, the conclusion is only pertinent to the sampled subjects. If the original labels were assigned by randomization (i.e., they represent an intervention) then the conclusion may claim causation.

4.1.1 Example 1: The Harris Bank Sex Pay Study

We have worked with this data before: there are 93 salaries of Harris Bank employees from 1969-1971. The workers are all entry-level clerical workers. We're interested in whether males and females are paid differently.

```
salaries.df <- read.csv('salaries.csv')
salaries.df
```

```
##    Salary Sex
## 1    4620   1
## 2    5040   1
## 3    5100   1
## 4    5100   1
## 5    5220   1
## 6    5400   1
## 7    5400   1
## 8    5400   1
## 9    5400   1
## 10   5400   1
## 11   5700   1
## 12   6000   1
## 13   6000   1
## 14   6000   1
## 15   6000   1
## 16   6000   1
## 17   6000   1
```

```
## 18   6000   1
## 19   6000   1
## 20   6000   1
## 21   6000   1
## 22   6000   1
## 23   6000   1
## 24   6000   1
## 25   6300   1
## 26   6600   1
## 27   6600   1
## 28   6600   1
## 29   6840   1
## 30   6900   1
## 31   6900   1
## 32   8100   1
## 33   3900   0
## 34   4020   0
## 35   4290   0
## 36   4380   0
## 37   4380   0
## 38   4380   0
## 39   4380   0
## 40   4380   0
## 41   4440   0
## 42   4500   0
## 43   4500   0
## 44   4620   0
## 45   4800   0
## 46   4800   0
## 47   4800   0
## 48   4800   0
## 49   4800   0
## 50   4800   0
## 51   4800   0
## 52   4800   0
## 53   4800   0
## 54   4800   0
## 55   4980   0
## 56   5100   0
## 57   5100   0
## 58   5100   0
## 59   5100   0
## 60   5100   0
## 61   5100   0
## 62   5160   0
## 63   5220   0
```

```
## 64   5220   0
## 65   5280   0
## 66   5280   0
## 67   5280   0
## 68   5400   0
## 69   5400   0
## 70   5400   0
## 71   5400   0
## 72   5400   0
## 73   5400   0
## 74   5400   0
## 75   5400   0
## 76   5400   0
## 77   5400   0
## 78   5400   0
## 79   5400   0
## 80   5520   0
## 81   5520   0
## 82   5580   0
## 83   5640   0
## 84   5700   0
## 85   5700   0
## 86   5700   0
## 87   5700   0
## 88   5700   0
## 89   6000   0
## 90   6000   0
## 91   6120   0
## 92   6300   0
## 93   6300   0
```

4.1.2 Difference in mean salaries between genders

```
n0 <- sum(salaries.df$Sex==0)
n1 <- sum(salaries.df$Sex==1)
m0 <- mean(salaries.df$Salary[salaries.df$Sex==0])
m1 <- mean(salaries.df$Salary[salaries.df$Sex==1])
s0 <- var(salaries.df$Salary[salaries.df$Sex==0])
s1 <- var(salaries.df$Salary[salaries.df$Sex==1])
t0 <- (m0 - m1)/sqrt(s0/n0+s1/n1)
c(t0,m0,m1,sqrt(s0),sqrt(s1))
```

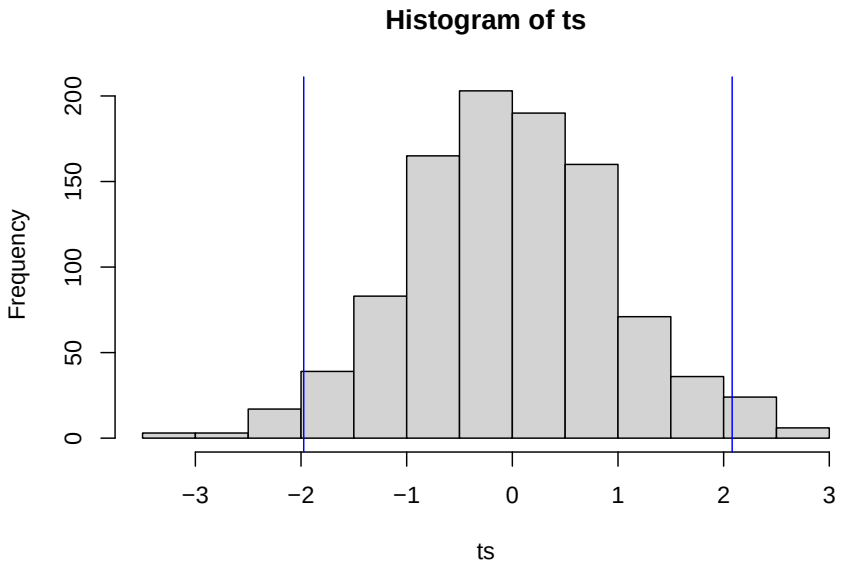
```
## [1]   -5.829974 5138.852459 5956.875000  539.870658  690.733306
```

4.1.3 Randomization test for treatment significance

Algorithm: for 1000 loops do: 1. randomly shuffle the sex labels over salaries 2. compute the two-sample t-test stat for the reshuffled labels return all 1000 differences

```
n0 <- sum(salaries.df$Sex==0)
n1 <- sum(salaries.df$Sex==1)
n <- n0+n1
randomization.test <- function(M){
  ts <- rep(NA,M)
  for(m in 1:M){
    permute.n <- sample.int(n,n,replace = F)
    re.randomized.group <- salaries.df$Sex[permute.n]
    m0 <- mean(salaries.df$Salary[re.randomized.group==0])
    m1 <- mean(salaries.df$Salary[re.randomized.group==1])
    s0 <- var(salaries.df$Salary[re.randomized.group==0])
    s1 <- var(salaries.df$Salary[re.randomized.group==1])
    ts[m] <- (m0 - m1)/sqrt(s0/n0+s1/n1)
    }
  return(ts)
}

ts <- randomization.test(1000)
hist(ts)
abline(v = quantile(ts, 0.975), col = 'blue')
abline(v = quantile(ts, 0.025), col = 'blue')
```



The t_0 value of -5.83 is off the chart (literally). How do we interpret this? The test randomly labels observed sexes with observed salaries. The observed test statistic suggests this is not how the salaries were actually assigned to workers. That's pretty obvious—companies do not tend to assign salaries in a random fashion. But, if the salary assignment was not random with respect to sex, then it either depended on sex or depends on a hidden variable closely associated with sex. Since the job, time frame, and experience level were controlled for, it is certainly plausible that the salary assignment depended on sex.

```
n0 <- sum(salaries.df$Sex==0)
n1 <- sum(salaries.df$Sex==1)
n <- n0+n1
randomization.test <- function(M, df){
  ts <- rep(NA,M)
  for(m in 1:M){
    permute.n <- sample.int(n,n,replace = F)
    re.randomized.group <- df$Sex[permute.n]
    m0 <- mean(df$Salary[re.randomized.group==0])
    m1 <- mean(df$Salary[re.randomized.group==1])
    s0 <- var(df$Salary[re.randomized.group==0])
    s1 <- var(df$Salary[re.randomized.group==1])
    ts[m] <- (m0 - m1)/sqrt(s0/n0+s1/n1)
    }
  return(ts)
}
```

```
grid.delta <- seq(from = 400, to = 1400, length.out = 100)
L <- length(grid.delta)
p.value <- rep(NA,L)
reject<-rep(NA, L)
for(j in 1:L){
temp.df <- salaries.df
temp.df$Salary[temp.df$Sex==1]<-temp.df$Salary[temp.df$Sex==1]-grid.delta[j]
m0 <- mean(temp.df$Salary[temp.df$Sex==0])
m1 <- mean(temp.df$Salary[temp.df$Sex==1])
s0 <- var(temp.df$Salary[temp.df$Sex==0])
s1 <- var(temp.df$Salary[temp.df$Sex==1])
t0 <- (m0 - m1)/sqrt(s0/n0+s1/n1)
test.vals.j <- randomization.test(10000, temp.df)
mean.randomization <- mean(test.vals.j)
p.value[j] <- ifelse(t0 > mean.randomization,sum(abs(test.vals.j)>t0)/10000,sum(-abs(test.vals.j)
}
cbind(grid.delta, p.value)
```

```
##        grid.delta p.value
##   [1,]   400.0000  0.0038
##   [2,]   410.1010  0.0040
##   [3,]   420.2020  0.0068
##   [4,]   430.3030  0.0078
##   [5,]   440.4040  0.0079
##   [6,]   450.5051  0.0106
##   [7,]   460.6061  0.0123
##   [8,]   470.7071  0.0142
##   [9,]   480.8081  0.0170
##  [10,]   490.9091  0.0188
##  [11,]   501.0101  0.0276
##  [12,]   511.1111  0.0317
##  [13,]   521.2121  0.0374
##  [14,]   531.3131  0.0415
##  [15,]   541.4141  0.0496
##  [16,]   551.5152  0.0598
##  [17,]   561.6162  0.0727
##  [18,]   571.7172  0.0823
##  [19,]   581.8182  0.0952
##  [20,]   591.9192  0.1132
##  [21,]   602.0202  0.1295
##  [22,]   612.1212  0.1484
##  [23,]   622.2222  0.1698
##  [24,]   632.3232  0.1932
```

```
##  [25,]   642.4242  0.2116
##  [26,]   652.5253  0.2535
##  [27,]   662.6263  0.2700
##  [28,]   672.7273  0.3006
##  [29,]   682.8283  0.3385
##  [30,]   692.9293  0.3814
##  [31,]   703.0303  0.4210
##  [32,]   713.1313  0.4567
##  [33,]   723.2323  0.5046
##  [34,]   733.3333  0.5444
##  [35,]   743.4343  0.6021
##  [36,]   753.5354  0.6418
##  [37,]   763.6364  0.7020
##  [38,]   773.7374  0.7513
##  [39,]   783.8384  0.8085
##  [40,]   793.9394  0.8648
##  [41,]   804.0404  0.9259
##  [42,]   814.1414  0.9766
##  [43,]   824.2424  0.9690
##  [44,]   834.3434  0.9066
##  [45,]   844.4444  0.8595
##  [46,]   854.5455  0.7955
##  [47,]   864.6465  0.7354
##  [48,]   874.7475  0.6922
##  [49,]   884.8485  0.6391
##  [50,]   894.9495  0.5835
##  [51,]   905.0505  0.5432
##  [52,]   915.1515  0.4895
##  [53,]   925.2525  0.4505
##  [54,]   935.3535  0.4126
##  [55,]   945.4545  0.3676
##  [56,]   955.5556  0.3285
##  [57,]   965.6566  0.2918
##  [58,]   975.7576  0.2609
##  [59,]   985.8586  0.2415
##  [60,]   995.9596  0.2065
##  [61,]  1006.0606  0.1829
##  [62,]  1016.1616  0.1658
##  [63,]  1026.2626  0.1415
##  [64,]  1036.3636  0.1237
##  [65,]  1046.4646  0.1110
##  [66,]  1056.5657  0.0925
##  [67,]  1066.6667  0.0838
##  [68,]  1076.7677  0.0673
##  [69,]  1086.8687  0.0598
##  [70,]  1096.9697  0.0507
```

```
##  [71,]  1107.0707  0.0433
##  [72,]  1117.1717  0.0384
##  [73,]  1127.2727  0.0308
##  [74,]  1137.3737  0.0267
##  [75,]  1147.4747  0.0229
##  [76,]  1157.5758  0.0191
##  [77,]  1167.6768  0.0167
##  [78,]  1177.7778  0.0117
##  [79,]  1187.8788  0.0097
##  [80,]  1197.9798  0.0082
##  [81,]  1208.0808  0.0073
##  [82,]  1218.1818  0.0070
##  [83,]  1228.2828  0.0047
##  [84,]  1238.3838  0.0036
##  [85,]  1248.4848  0.0038
##  [86,]  1258.5859  0.0026
##  [87,]  1268.6869  0.0017
##  [88,]  1278.7879  0.0016
##  [89,]  1288.8889  0.0010
##  [90,]  1298.9899  0.0011
##  [91,]  1309.0909  0.0013
##  [92,]  1319.1919  0.0004
##  [93,]  1329.2929  0.0004
##  [94,]  1339.3939  0.0005
##  [95,]  1349.4949  0.0005
##  [96,]  1359.5960  0.0003
##  [97,]  1369.6970  0.0001
##  [98,]  1379.7980  0.0002
##  [99,]  1389.8990  0.0001
## [100,]  1400.0000  0.0001
```

```
midpoint <- which.max(p.value)
lower.endpoint <- grid.delta[which.min(abs(0.05-p.value[1:midpoint]))]
upper.endpoint <- grid.delta[midpoint+which.min(abs(0.05-p.value[(midpoint+1):L]))]
c(lower.endpoint, upper.endpoint)
```

```
## [1]  541.4141 1096.9697
```

Chapter 5

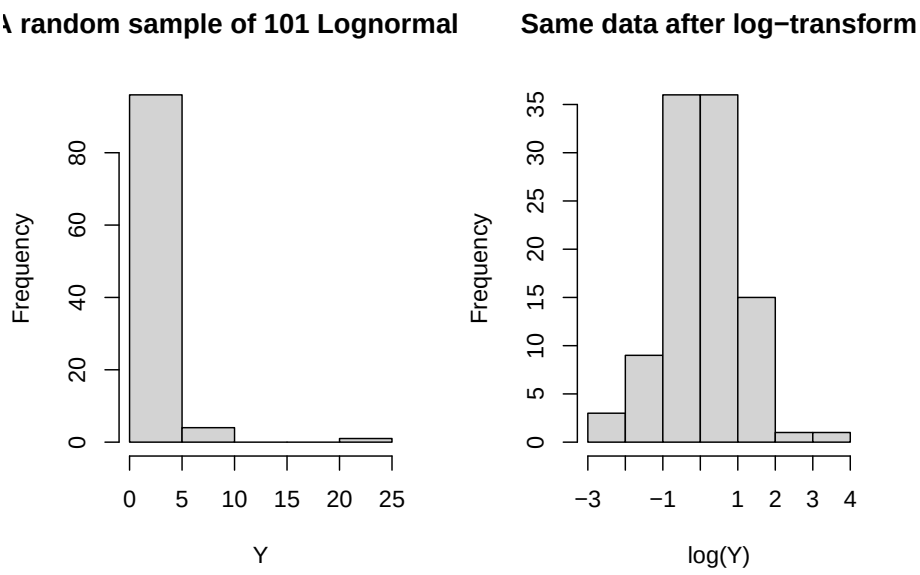
Alternatives for Non-Normal Responses

When responses (or residuals) appear non-normal we may doubt the validity of z-, t-, and F- tests, especially for small sample sizes. We'll discuss two strategies for dealing with non-normality: transformations and robust tests.

5.1 Transformations

Sometimes, when residuals/responses appear non-normal, a particular transformation/function of the values will produce residuals/responses that do appear normal. The case-in-point is the Lognormal distribution. It is, essentially, defined by the following relationship: If X ~Lognormal, then $\log(X)$ ~Normal. And, vice-versa, if Y ~Normal, then $\exp(Y)$ ~Lognormal. A random sample from a lognormal distribution will appear skewed to the right, or positively skewed. The log transformation dampens large (absolute) values more than small ones, reducing or removing the skew.

```
Y <- exp(rnorm(101))
par(mfrow = c(1,2))
hist(Y, main = 'A random sample of 101 Lognormal r.v. s')
hist(log(Y), main = 'Same data after log-transform')
```




```
median(Y)
```

```
## [1] 1.037574
```

```
median(log(Y))
```

```
## [1] 0.03688552
```

```
log(median(Y))
```

```
## [1] 0.03688552
```

Of course, the log-transform can only be applied to positive data values, and is meant specifically for use with positively skewed data. Since the log transform is monotonic increasing it preserves medians. That is, the median of the log-transformed values is equal to the log of the median of the original values. This is helpful for interpreting the results of tests concerning the mean of the log-transformed values.

Allow me to elaborate. Consider a one-sample test of $H_0 : \mu = \mu_0$ where μ denotes the population mean of the log-transformed random variable $X = \log(Y)$. Assuming X follows a normal distribution, its mean and median are

the same. So, if we reject the hypothesis and say $\mu \neq \mu_0$ this is equivalent to saying the **median** of the Y population is not $\log(\mu_0)$.

For a two sample test of $H_0 : \mu_1 = \mu_2$ the point estimate $\overline{X}_1 - \overline{X}_2$ measures the difference in population means for the log-transformed random variables, and $\exp(\overline{X}_1 - \overline{X}_2)$ estimates the **ratio of medians** for the populations on the original scale.

The log transform is not the only transformation that may aid in "normalizing" right-skewed responses. Sometimes a square root transformation is appropriate. A general method is available to select the "best" normalizing transformation for right-skewed data. I won't elaborate on this method here, but simply note that in R the function "boxcox" will determine a good transformation to use to normalize right-skewed responses. These transformations are the Box-Cox family of functions

$$\frac{y^\gamma - 1}{\gamma} \quad \gamma \neq 0,$$

and where the transformation is defined to be the log-transform when $\gamma = 0$.

5.1.1 Example: Cloud Seeding for Rainfall

The data frame "clouds" in the HSAUR2 package contains rainfall measurements (among other things) when researchers have "seeded" clouds in the atmosphere compared to when they have not done so. The data is noticeably right-skewed, and, of course positive because we're talking about rainfall measurments.

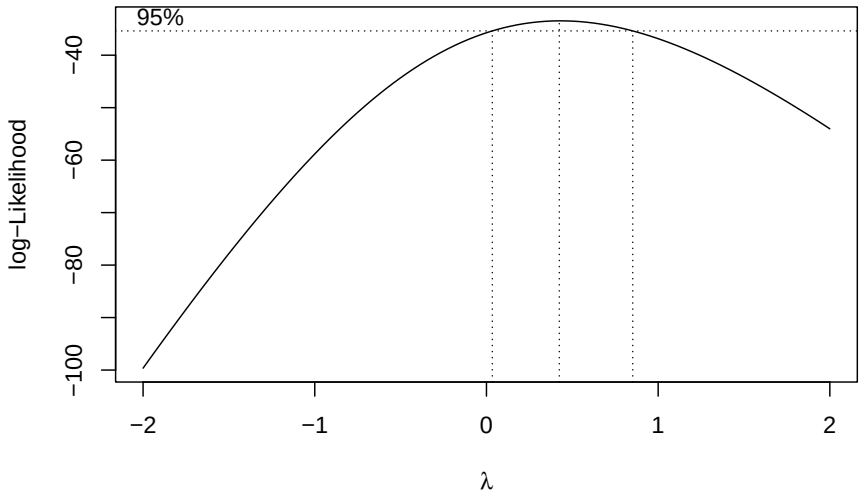
```
library(HSAUR2)
```

```
## Loading required package: tools
```

```
boxplot(rainfall ~ seeding, data = clouds, ylab = "Rainfall")
```

Using the boxcox function we see that the "best" normalizing transformation has $\gamma \approx 0.42$. Briefly, the "best" transformation is the one maximizing a normal likelihood. And, indeed boxplots of rainfall measurements under this transformation appear closer to normal.

```
library(MASS)
bc <- boxcox(rainfall~seeding, data = clouds)
```




```
bc$x[which.max(bc$y)]
```

```
## [1] 0.4242424
```

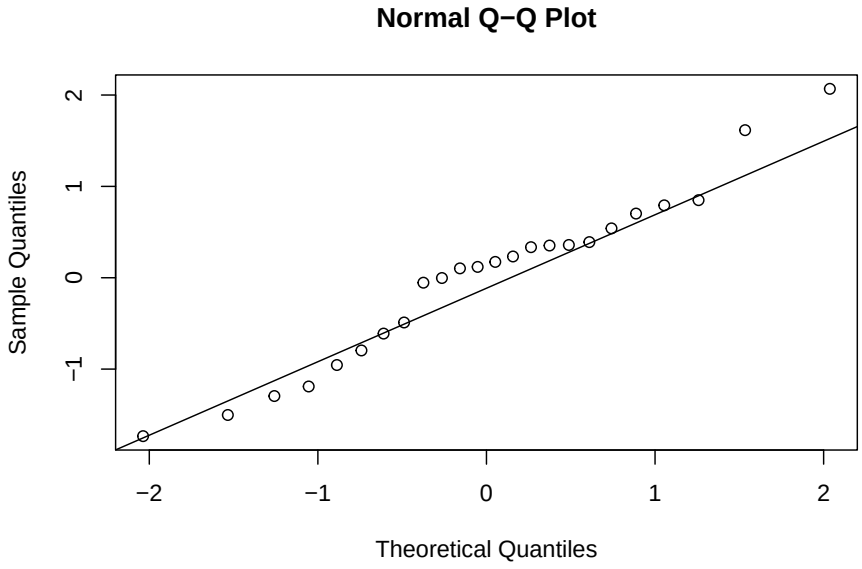
```
my.clouds <- clouds
my.clouds$bc <- (my.clouds$rainfall^0.42 - 1)/ 0.42
boxplot(bc~seeding, data = my.clouds)
```

Let's do a two-sample t-test for difference in mean rainfall under seeding/non-seeding conditions. Remember to interpret the results carefully due to our transformation.

```
bc.t.test <- t.test(my.clouds$bc[my.clouds$seeding == 'no'], my.clouds$bc[my.clouds$see
bc.t.test
```

```
##
##  Welch Two Sample t-test
##
## data:  my.clouds$bc[my.clouds$seeding == "no"] and my.clouds$bc[my.clouds$seeding ==
## t = -0.7541, df = 19.094, p-value = 0.46
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.6280579  0.7654181
## sample estimates:
## mean of x mean of y
##  1.551329  1.982649
```

```
n1 <- sum(my.clouds$seeding == 'no')
n2 <- sum(my.clouds$seeding == 'yes')
sp2 <- (n1*var(my.clouds$bc[my.clouds$seeding == 'no']) + n2*var(my.clouds$bc[my.cloud
residuals <- c( my.clouds$bc[my.clouds$seeding == 'no'] - mean(my.clouds$bc[my.clouds$
qqnorm(residuals)
qqline(residuals)
```



We do not reject the hypothesis of equal mean (transformed) rainfall. On the original data scale, we say the ratio of population median rainfall amounts is not significantly different than 1.

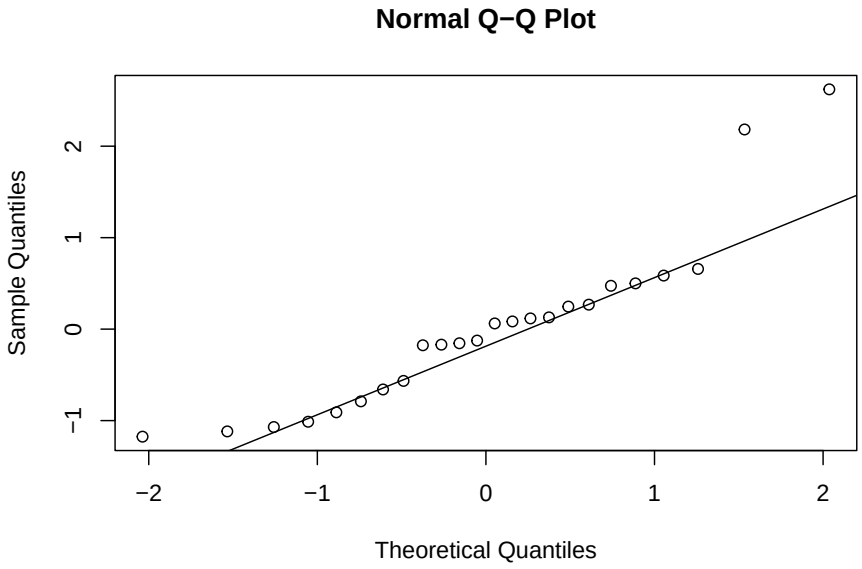
We can look also at the t-test on the original data scale and see to what extent the residuals may suggest the normality assumption is suspect.

```
orig.t.test <- t.test(clouds$rainfall[my.clouds$seeding == 'no'], clouds$rainfall[clouds$seeding
orig.t.test
```

```
##
##  Welch Two Sample t-test
##
## data:  clouds$rainfall[my.clouds$seeding == "no"] and clouds$rainfall[clouds$seeding == "yes"]
## t = -0.3574, df = 20.871, p-value = 0.7244
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3.154691  2.229691
## sample estimates:
## mean of x mean of y
##  4.171667  4.634167
```

```
n1 <- sum(clouds$seeding == 'no')
n2 <- sum(clouds$seeding == 'yes')
sp2 <- (n1*var(clouds$rainfall[clouds$seeding == 'no']) + n2*var(clouds$rainfall[clouds$seeding =
```

```
residuals <- c( clouds$rainfall[clouds$seeding == 'no'] - mean(clouds$rainfall[clouds$s
qqnorm(residuals)
qqline(residuals)
```



At least the tails of the residuals certainly look like less of a match than the Box-Cox transformed responses. This is consistent with the strong right skew we see in the original rainfall data.

5.2 Rank-Sum Test

We have seen that certain one-to-one transformations can be helpful when dealing with positively-skewed responses. But, what about other types of non-normal data? One general-purpose approach to comparing two populations of non-normal responses is to apply a rank transformation and use the Mann-Whitney (Rank-Sum) test.

Unlike the previous Box-Cox transformations, the rank transformation is not one-to-one, so some information is lost moving from the original to the transformed data (the ranks). But, in exchange for this price we pay we obtain a data set that is easier to analyze, especially in certain special cases.

To perform the rank-sum test we first sort all the responses, with the two groups pooled together. If some responses have the same value, we assign to them the average of the corresponding ranks. For example, the sorted responses 1,2,2,3,4,6

would get the ranks 1,2.5,2.5,4,5,6. Let $T = \max\{T_1, T_2\}$ where T_1 and T_2 are the sums of the ranks for responses in groups 1 and 2, respectively. Let n_T be the sample size of the group corresponding to T. Let $\overline{r}$ and s_r be the sample mean and sample standard deviation of all of the ranks. Then, the rank-sum test statistic is defined as

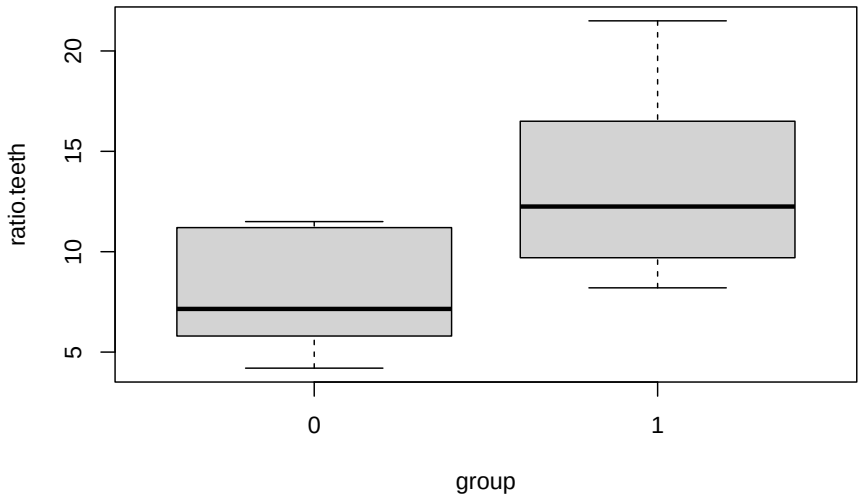
$$z = \frac{T - n_T \overline{r}}{s_r \sqrt{\frac{n_1 n_2}{n_1 + n_2}}}.$$

For large n_1 and n_2, $z \dot{\sim} N(0, 1)$, so the test conclusion follows from comparing z to standard normal quantiles. For example, the p-value is $2(1 - \Phi(z))$.

Just what does the rank-sum test test? In the most general form, under no further assumptions, the rank-sum tests equality of distributions of the response in the two groups. However, the test will only detect differences in those distributions if $P(X_1 > X_2) \neq P(X_2 > X_1)$ where X_1 and x_2 denote the response random variable from each population. For example, if the populations are both normal with the same mean but different variances, then the rank-sum test will not generally be able to distinguish between the populations. In the special case that the populations are different by a location shift, i.e., $F_1(x) = F_1(x + c)$ for a constant c, the null hypothesis is equivalent to testing equality of population medians.

5.2.1 Example: ratio measurements

The following data represent the ratios of lengths of canine and molar teeth in two populations of animals: wild and captive. Ratios are tricky, and may behave badly when the denominator can be close to zero. These ratio measurements exhibit non-normality, but not positive skew like we have seen before.



Next, we'll implement the rank-sum test.

```
teeth <- teeth[order(teeth$ratio.teeth),]

ranks <- rank(teeth$ratio.teeth, ties = 'average')

teeth$ranks <- ranks

T1 <- sum(teeth$ranks[teeth$group == 1])
T0 <- sum(teeth$ranks[teeth$group == 0])
T <- max(T1, T0)
n1 <- sum(teeth$group == 1)
n0 <- sum(teeth$group == 0)
nt <- ifelse(T==T1, n1, n0)
rbar <- mean(teeth$ranks)
sr <- sd(teeth$ranks)

test.stat <- (T - nt*rbar)/(sr * sqrt(n1*n0/(n1+n0)))
test.stat
```

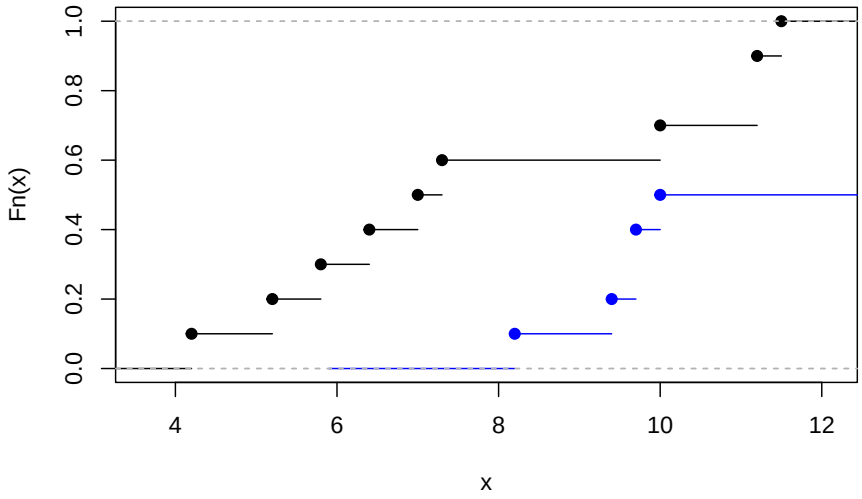
```
## [1] 2.308188
```

```
2*(1-pnorm(test.stat))
```

```
## [1] 0.02098868
```

The p-value is less than 0.05 so we would reject the null hypothesis of equal distributions. The teeth measurements have different distributions in the captive and wild populations. Digging in a bit further, we plot the empirical distribution functions of the measurements in each group. This shows the CDFs are approximately equivalent up to a location shift. So, it is reasonable to interpret the rank-sum test and rejecting the hypothesis of equal medians: the wild and captive populations have different median tooth ratios.

```
F0 <- ecdf(teeth$ratio.teeth[teeth$group==0])
F1 <- ecdf(teeth$ratio.teeth[teeth$group==1])
plot(F0, main = '')
lines(F1, col = 'blue')
```



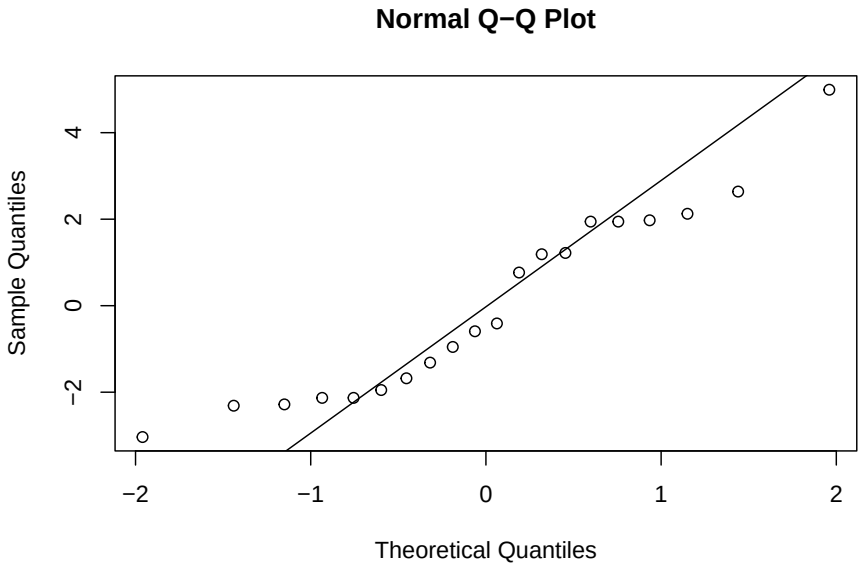
Finally, compare the the rank-sum test to Student's t-test. The t-test shows an even smaller p-value, but note that the qq-plot shows strong non-normality, which calls into question the appropriateness of the t-test.

```
my.t.test <- t.test(ratio.teeth~group, data = teeth)
my.t.test
```

```
##
##  Welch Two Sample t-test
##
## data:  ratio.teeth by group
```

```
## t = -3.1697, df = 14.969, p-value = 0.006359
## alternative hypothesis: true difference in means between group 0 and group 1 is not
## 95 percent confidence interval:
##  -8.780943 -1.719057
## sample estimates:
## mean in group 0 mean in group 1
##            7.98           13.23
```

```
z <- c((teeth$ratio.teeth[teeth$group==0] - my.t.test$estimate[1]) / my.t.test$stderr,
qqnorm(z)
qqline(z)
```



5.3 Signed-rank test

The signed-rank test or Wilcoxon signed-rank test is the paired analog of the Mann-Whitney test. As an illustrative example we consider a real twins study (not simulated this time) comparing BMIs in pairs of twins.

```
library(mets)
```

```
## Loading required package: timereg
```

```
## Loading required package: survival
```

```
## Loading required package: lava
```

```
## mets version 1.3.0
```

```
data("twinbmi")
head(twinbmi)
```

```
##   tvparnr      bmi      age gender zyg id num
## 1       1 26.33289 57.51212   male  DZ  1   1
## 2       1 25.46939 57.51212   male  DZ  1   2
## 3       2 28.65014 56.62696   male  MZ  2   1
## 5       3 28.40909 57.73097   male  DZ  3   1
## 7       4 27.25089 53.68683   male  DZ  4   1
## 8       4 28.07504 53.68683   male  DZ  4   2
```

```
twinwide <- fast.reshape(twinbmi, id="tvparnr",varying=c("bmi"))
head(twinwide)
```

```
##    tvparnr     bmi1      age gender zyg id num     bmi2
## 1        1 26.33289 57.51212   male  DZ  1   1 25.46939
## 3        2 28.65014 56.62696   male  MZ  2   1       NA
## 5        3 28.40909 57.73097   male  DZ  3   1       NA
## 7        4 27.25089 53.68683   male  DZ  4   1 28.07504
## 9        5 27.77778 52.55838   male  DZ  5   1       NA
## 11       6 28.04282 52.52231   male  DZ  6   1 22.30936
```

```
twinwide$bmidiff <- (twinwide$bmi1 - twinwide$bmi2)
twinwide.cc <- complete.cases(twinwide)
twinwide <- twinwide[twinwide.cc,]
head(twinwide)
```

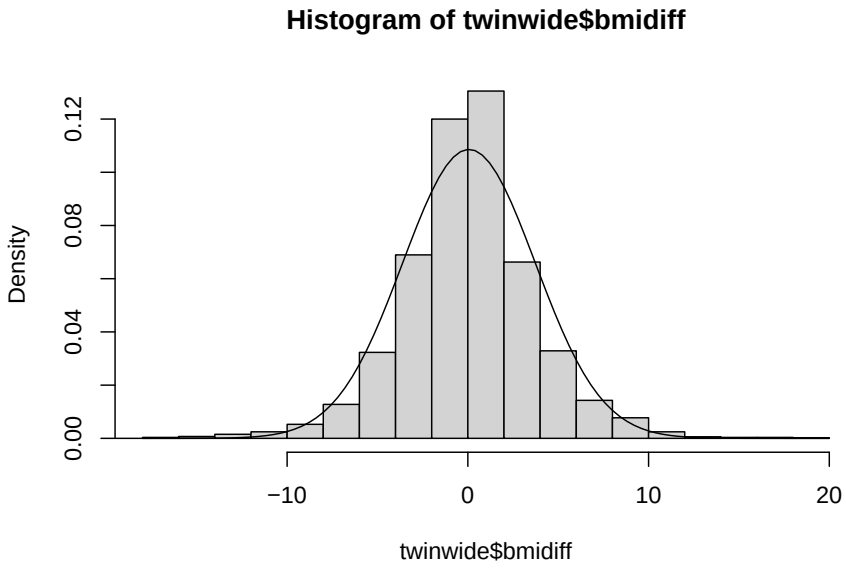
```
##    tvparnr     bmi1      age gender zyg id num     bmi2  bmidiff
## 1        1 26.33289 57.51212   male  DZ  1   1 25.46939  0.86350
## 7        4 27.25089 53.68683   male  DZ  4   1 28.07504 -0.82415
## 11       6 28.04282 52.52231   male  DZ  6   1 22.30936  5.73346
## 13       7 28.06642 52.62944   male  DZ  7   1 26.51180  1.55462
## 19      10 30.47797 51.24806   male  DZ 10   1 27.66010  2.81787
## 23      12 27.39818 50.02067   male  DZ 12   1 25.97012  1.42806
```

```
dim(twinwide)
```

```
## [1] 4271    9
```

The BMI differences may appear normal but actually exhibit *kurtosis*. Since this is an unusual non-normality we will apply a rank transformation.

```
hist(twinwide$bmidiff, freq = F)
dnorm.tw <- function(x) dnorm(x,mean(twinwide$bmidiff), sd(twinwide$bmidiff))
curve(dnorm.tw, -20,20, add = TRUE)
```



We can compute the test statistic and p-value by hand as follows: let T be the sum of the ranks of the absolute BMI differences in the group of positive differences. Then, $z = (T - \frac{1}{4}n(n+1))/\sqrt{n(n+1)(2n+1)/24}$ is approximately standard normal.

```
absdiff <- abs(twinwide$bmidiff)
signs <- ifelse(twinwide$bmidiff<0,-1,1)
n<-length(signs)
sr.nums <- cbind(absdiff, signs, rank(absdiff))
sr.nums <- sr.nums[order(sr.nums[,1]),]
T <- sum(sr.nums[sr.nums[,2]==1,3])
z <- (T - n*(n+1)/4)/sqrt(n*(n+1)*(2*n+1)/24)
2*(1-pnorm(abs(z)))
```

```
## [1] 0.1904966
```

R has built in functions for the rank-sum and signed-rank tests.

```
wilcox.test(twinwide$bmi1, twinwide$bmi2, paired = TRUE)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  twinwide$bmi1 and twinwide$bmi2
## V = 4551983, p-value = 0.2021
## alternative hypothesis: true location shift is not equal to 0
```

In this case we likely will not come to a different conclusion if we apply Student's t-test.

```
t.test(twinwide$bmi1, twinwide$bmi2, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  twinwide$bmi1 and twinwide$bmi2
## t = 1.1008, df = 4270, p-value = 0.2711
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -0.04832046  0.17204938
## sample estimates:
## mean difference
##      0.06186446
```

5.4 Bootstrap

5.4.1 Intro to the bootstrap

For hypothesis testing and confidence intervals, there is a "statistic" whose sampling distribution is required

For example, to test $H_0 : \mu = \mu_0$ using a normal random sample $X_1, ..., X_n$ with $X_i \overset{iid}{\sim} N(\mu, \sigma^2)$ the test statistic

$$T = \frac{\bar{X} - \mu_0}{S/\sqrt{n}}$$

has a t distribution with $n-1$ degrees of freedom given the null hypothesis is true.

Only in very simple problems can we determine an "exact" test statistic. For example, in the two-sample t-test with unknown, unequal population variances, the T statistic is only approximate.

We often rely on assumptions (like normality) or large sample sizes (CLT) in order to derive approximate (or asymptotic) distributions of statistics.

The bootstrap (in a manner similar to Monte Carlo) tries to approximate the distribution of a test statistic using simulations.

For a distribution with cdf F, we are interested in a parameter $\theta = \phi(F)$, and functional of F.

Examples: 1. Mean - $\phi(F) = \int x dF(x)$ 2. Anything else that is a function of moments, like variance 3. Median - $\phi(F) = \inf\{x : F(x) \geq 0.5\}$…

Given data $X_1, ..., X_n$ the empirical CDF (think of this as an estimate of F) is

$$\hat{F}(x) = \frac{1}{n} \sum_{i=1}^{n} 1\{X_i \leq x\}.$$

A natural estimate of θ is the "plug-in" estimator $\hat{\theta} = \phi(\hat{F})$.

Given the previous setup of CDF and empirical CDF, suppose we want to estimate θ. If we could somehow simulate from F, then our knowledge of Monte Carlo suggests the following: 1. Simulate many data sets, say M, of size n: $X_{1,1}, ..., X_{n,1}$, …, $X_{1,M}, ..., X_{n,M}$. 2. From the M data sets, compute $\hat{\theta}_i$ for $i = 1, ..., M$. 3. Then, our many estimates should given us an idea of the mean and the variance of $\hat{\theta}$.

This actually works when we know the distribution of the test statistic. In the one-sample t-test for a normal mean, the test statistic has a $t(n-1)$ distribution. So, if we sample from this distribution many times, we will be able to find the rejection region for the hypothesis test by computing Monte Carlo estimates of the $t(n-1)$ quantiles, but this is rather trivial…

Of course, we (usually) cannot simulate from F because it is unknown (depends on unknown parameters).

Bootstrap idea: Replace simulation from F with simulation from $\hat{F}$!

Note: it is easy to simulate from $\hat{F}$, it's just simulation from a discrete uniform distribution with probabilities $1/n$ on each data point.

Suppose we are given the following two samples:

```
set.seed(3214354)
x <- rexp(10,1)
y <- rexp(9,0.8)
round(x,2)
```

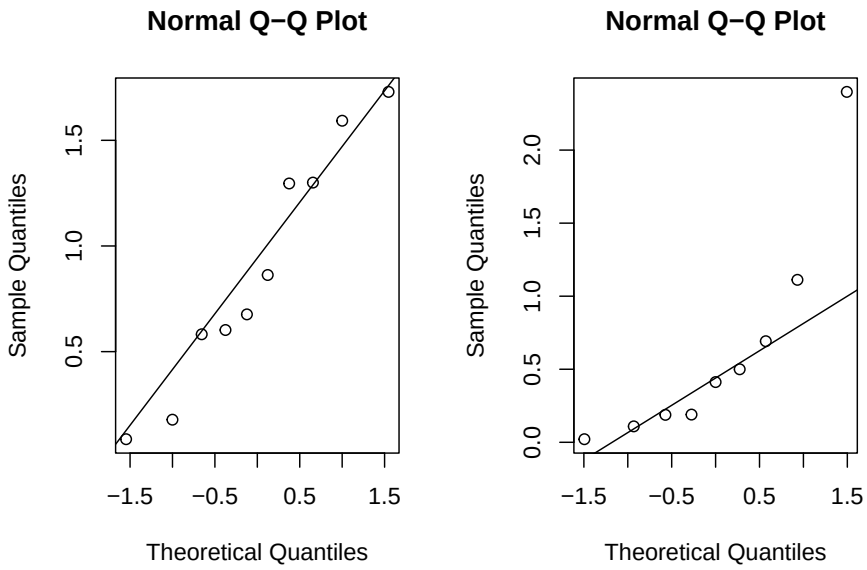
```
##  [1] 0.18 1.30 0.09 1.30 1.73 0.86 1.59 0.60 0.58 0.68
```

```
round(y,2)
```

```
## [1] 0.02 0.50 0.41 0.19 0.19 0.69 2.40 1.11 0.11
```

We could use the t-test, but wait...

```
par(mfrow = c(1,2))
qqnorm(x)
qqline(x)
qqnorm(y)
qqline(y)
```




```
shapiro.test(x)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x
## W = 0.94146, p-value = 0.5694
```

```
shapiro.test(y)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  y
## W = 0.77085, p-value = 0.009449
```

What "test statistic" should I use to test $H_0 : \mu_X = \mu_Y$? Why not $\bar{X} - \bar{Y}$?

```
test.statistic <- mean(x) - mean(y)
boot.test.statistic <- rep(NA, 10000)
for(i in 1:10000){
  boot.x <- sample(x,replace = T)
  boot.y <- sample(y, replace = T)
  boot.test.statistic[i] <- mean(boot.x)-mean(boot.y)
}
test.statistic
```

```
## [1] 0.2657391
```

```
quantile(boot.test.statistic,0.025)
```

```
##      2.5%
## -0.345881
```

```
quantile(boot.test.statistic,0.975)
```

```
##    97.5%
## 0.792484
```

```
m <- median(boot.test.statistic)

approx.p.value = ifelse(test.statistic > m, sum((boot.test.statistic-m) > (test.statist
approx.p.value
```

```
## [1] 0.9726
```

5.4.2 Bootstrap Confidence Intervals

A variety of methods are used to construct confidence intervals using the bootstrap distribution of an estimator. We'll discuss the percentile, basic, and Studentized methods, although several others are available, including the bias-corrected and accelerated (BCa) intervals computed by default in R's boot package.

Suppose the bootstrap samples $\hat{\theta}_1, \ldots, \hat{\theta}_B$ are available. The simplest bootstrap interval estimate is given by the percentile method, which defines the $100(1-\alpha)\%$ CI to be the $\alpha/2$ to $(1-\alpha/2)$ sample quantiles of $\hat{\theta}_1, \ldots, \hat{\theta}_B$. There are good arguments against the percentile method, but we won't wade into the details presently. Those arguments tend to favor the basic bootstrap method, which defines the $100(1-\alpha)\%$ CI to be the interval $(2\overline{\theta} - \hat{\theta}_{1-\alpha/2}, 2\overline{\theta} - \hat{\theta}_{\alpha/2})$, which is twise the sample mean of the bootstrapped estimates minus the $(1-\alpha/2)$ and $\alpha/2$ sample quantiles of those bootstrapped estimates $\hat{\theta}_1, \ldots, \hat{\theta}_B$.

The last method we'll discuss is the Studentized bootstrap. The idea of the method is to mimic the usual z- or t-test pivotal quantity $\frac{\hat{\theta}-\theta}{se(\hat{\theta})}$. First, compute $\hat{\theta}$ for teh original data. Then, perform the following nested bootstrap, for b in 1 to B do: 1.Resample the data with replacement to generate the b^{th} bootstrap data set a. for this b^{th} data set compute the point estimate $\tilde{\theta}_b$ b. Resample the resamples! Generate K nested bootstrap resamples sample of data. For each of them compute $\tilde{\theta}_{b,k}$ for $k = 1, \ldots, K$. c. Estimate $se(\tilde{\theta})$ as the standard deviation of $\tilde{\theta}_{b,k}$ for $k = 1, \ldots, K$. d. Keep the studentized value $q_b = \frac{\tilde{\theta}_b - \hat{\theta}}{se(\tilde{\theta}_b)}$ 2. Let $se(\hat{\theta})$ be estimated by the sample standard deviation of $\tilde{\theta}_1, \ldots, \tilde{\theta}_B$. 3. Define the Studentized interval as

$$(\hat{\theta} - q_{1-\alpha/2} se(\hat{\theta}),\ \hat{\theta} - q_{\alpha/2} se(\hat{\theta}))$$

where q_α is the α sample quantile of the Studentized q values.

```
set.seed(3214354)
x <- rexp(10,1)
y <- rexp(9,0.8)
round(x,2)
```

```
##  [1] 0.18 1.30 0.09 1.30 1.73 0.86 1.59 0.60 0.58 0.68
```

```
round(y,2)
```

```
## [1] 0.02 0.50 0.41 0.19 0.19 0.69 2.40 1.11 0.11
```

```
my.boot <- function(x,y, B, K, alpha){
  hattheta <- mean(x) - mean(y)
  tildetheta <- rep(NA,B)
  q <- rep(NA,B)
  for(b in 1:B){
  indices1 <- sample(10,10,replace = TRUE)
  indices2 <- sample(9,9,replace = TRUE)
  xstar <- x[indices1]
  ystar <- y[indices2]
  tildetheta[b] <- mean(xstar) - mean(ystar)
  tilde2 <- rep(NA,K)
    for(k in 1:K){
        indices1 <- sample(10,10,replace = TRUE)
        indices2 <- sample(9,9,replace = TRUE)
        xstar2 <- xstar[indices1]
        ystar2 <- ystar[indices2]
        tilde2[k] <- mean(xstar2)-mean(ystar2)
    }
  q[b] <- (tildetheta[b] - hattheta)/sd(tilde2)
  }
  intv <- hattheta - sd(tildetheta)*quantile(q,c(1-alpha/2,alpha/2))
  return(as.numeric(intv))
}
my.boot(x,y,1000,50,0.05)
```

```
## [1] -0.6147748  0.8576096
```

5.4.3 Bootstrapping linear models

When we have only one sample of iid data, bootstrapping is straightforward: simply sample with replacement n times from the original data to obtain a single bootstrap-resampled data set. Repeat many times, recording the estimate corresponding to each resampled set. When there is more "structure" present in the data we need to think carefully about how that structure should be treated by the resampling procedure.

In linear models, i.e., the Gauss-Markov model, for each response Y_i there is a corresponding covariate vector X_i. How should these pairs (Y_i, X_i) be treated by the resampling procedure? One method for bootstrapping a linear model is called row-resampling. Imagine binding the $n \times 1$ column of responses Y to the $n \times p$ matrix of covariates X and then resampling rows with replacement. We can do this by sampling the integers $1, \ldots, n$ with replacement, each time taking the corresponding row of (Y, X) as the bootstrap resample. Row resampling treats the pairs (Y_i, X_i) as random, but this is not always reasonable. For example, in a two-sample experiment in which we apply a treatment to 10 of 20 subjects

at random and keep the remaining 10 as a control group, row resampling will produce resampled data sets with more than ten in one of the two groups and fewer in the other, which is not how the experiment was performed. When the *design* is fixed, e.g., we have precisely ten subjects in each group, a different bootstrap method, called residual boostrap, makes more sense. The residual bootstrap starts with the residuals $e = y - \hat{y} = y - X_i\hat{\beta}$. We resample the residuals to obtain $e^\star = (e_1^\star, \dots, e_n^\star)^\top$. Next, we compute the bootstrapped responses, $y_i^\star = X_i\hat{\beta} + e_i^\star$. Finally, we compute $\hat{\beta}^\star$ using X and $y^\star$. This way, the design matrix X remains the same throughout the bootstrap procedure, preserving the structure of the experiment. Example: Consider the following experiment to evaluate the effect of a bacterial infection on the lifespan of Guinea pigs as measure in days. First, we apply the ro-resampling method. Admittedly, this method makes little sense if we intend for the experiment to have 64 pigs in the control group and 58 in the treatment group.

```
pigs <- read.csv('guinea_pigs.csv')
my.lm <- lm(Time~Treatment, data = pigs)
my.lm
```

```
##
## Call:
## lm(formula = Time ~ Treatment, data = pigs)
##
## Coefficients:
##      (Intercept)  TreatmentControl
##            242.5             102.7
```

```
my.lm$coefficients
```

```
##      (Intercept) TreatmentControl
##         242.5345         102.6843
```

```
n <- length(pigs$Pig)

r.r.coefs <- matrix(NA,1000,2)
row.resample.boot <- function(indices){
  data.star <- pigs[indices,]
  my.lm <- lm(Time~Treatment, data = data.star)
  return(as.numeric(my.lm$coefficients))
}
for(i in 1:1000){
  r.r.coefs[i,] <- row.resample.boot(sample.int(n,n,TRUE))
}
r.r.means <- colMeans(r.r.coefs)
```

```
percentiles1 <- quantile(r.r.coefs[,1], c(0.05,0.95))
percentiles1
```

```
##       5%      95%
## 218.0000 268.3603
```

```
percentiles2 <- quantile(r.r.coefs[,2], c(0.05,0.95))
percentiles2
```

```
##        5%       95%
##  49.40079 154.59927
```

```
c(2*r.r.means[1] - percentiles1[2], 2*r.r.means[1] - percentiles1[1])
```

```
##      95%       5%
## 216.4051 266.7653
```

```
c(2*r.r.means[2] - percentiles2[2], 2*r.r.means[2] - percentiles2[1])
```

```
##       95%        5%
##  50.48009 155.67856
```

Next, we implement the residual-bootstrap method. Note the two methods give very similar 90% basic bootstrap CIs for β_2 but disagree a bit for β_1.

```
pigs <-
read.csv('guinea_pigs.csv')
my.lm <- lm(Time~Treatment, data = pigs)
my.lm
```

```
##
## Call:
## lm(formula = Time ~ Treatment, data = pigs)
##
## Coefficients:
##      (Intercept)  TreatmentControl
##            242.5             102.7
```

```
my.lm$coefficients
```

```
##      (Intercept) TreatmentControl
##         242.5345         102.6843
```

```
n <- length(pigs$Pig)
resids <- my.lm$residuals

r.r.coefs <- matrix(NA,1000,2)
resid.boot <- function(indices){
  resids.star <- resids[indices]
  y.star <- cbind(rep(1,n), ifelse(pigs$Treatment=="Control",1,0))%*%matrix(as.numeric(my.lm$coef
  data.star <- pigs
  data.star$y.star <- y.star
  my.lm <- lm(y.star~Treatment, data = data.star)
  return(as.numeric(my.lm$coefficients))
}
for(i in 1:1000){
  r.r.coefs[i,] <- resid.boot(sample.int(n,n,TRUE))
}
r.r.means <- colMeans(r.r.coefs)
percentiles1 <- quantile(r.r.coefs[,1], c(0.05,0.95))
percentiles1
```

```
##       5%      95%
## 205.1938 281.0952
```

```
percentiles2 <- quantile(r.r.coefs[,2], c(0.05,0.95))
percentiles2
```

```
##        5%       95%
##  50.20677 154.69151
```

```
c(2*r.r.means[1] - percentiles1[2], 2*r.r.means[1] - percentiles1[1])
```

```
##      95%       5%
## 203.5823 279.4838
```

```
c(2*r.r.means[2] - percentiles2[2], 2*r.r.means[2] - percentiles2[1])
```

```
##       95%        5%
##  52.21648 156.70123
```

Chapter 6

One Way ANOVA

6.1 The Gauss-Markov Model for comparing I populations

As we saw in the second chapter, the Gauss-Markov linear model easily extends to comparisons of $I > 2$ populations using either a means or effects parametrization.

In this chapter, we discuss how to compare I populations efficiently, follow-up tests when not all populations are the same, and how to check model assumptions.

The model we will use throughout is

$$Y = X\beta + \epsilon$$

where Y is the vector of sampled responses for each of I groups/populations, X encodes the group membership structure of the responses, β is an $I \times 1$ vector that encodes the population means μ_i, $i = 1, \ldots, I$ (not necessarily by the identity function), and ϵ is a vector of iid Gaussian random residuals with homogeneous variance σ^2.

6.1.1 Example: Donuts effects model

Throughout this chapter we will consider the donuts data: there are 24 observations of oil absorption (in grams above 150 grams) for 4 types of oil (6 replicates per oil, balanced).

```
donuts <- read.csv('donuts.csv')
head(donuts)
```

```
##   Oil  y
## 1   1 14
## 2   1 22
## 3   1 18
## 4   1 27
## 5   1  6
## 6   1 45
```

```
donuts$Oil <- factor(donuts$Oil, levels = unique(donuts$Oil))
```

The default R parametrization is an effects model where β_1 is the Oil level 1 mean response, and β_2 through β_4 are differences from baseline, i.e., $\beta_2 = \mu_2 - \mu_1$.

```
my.lm<-lm(y~Oil, data = donuts)
model.matrix(my.lm)
```

```
##    (Intercept) Oil2 Oil3 Oil4
## 1            1    0    0    0
## 2            1    0    0    0
## 3            1    0    0    0
## 4            1    0    0    0
## 5            1    0    0    0
## 6            1    0    0    0
## 7            1    1    0    0
## 8            1    1    0    0
## 9            1    1    0    0
## 10           1    1    0    0
## 11           1    1    0    0
## 12           1    1    0    0
## 13           1    0    1    0
## 14           1    0    1    0
## 15           1    0    1    0
## 16           1    0    1    0
## 17           1    0    1    0
## 18           1    0    1    0
## 19           1    0    0    1
## 20           1    0    0    1
## 21           1    0    0    1
## 22           1    0    0    1
## 23           1    0    0    1
```

```
## 24           1    0    0    1
## attr(,"assign")
## [1] 0 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$Oil
## [1] "contr.treatment"
```

6.2 Testing equality of means

The ANOVA null hypothesis is equalit of all population means $H_0 : \mu_1 = \mu_2 = \cdots = \mu_I$ versus H_a : not all means are equal. Given the other assumptions (normality and homogeneous variance) the null hypothesis implies equality of all I populations.

Somewhat counterintuitively, the key to efficiently testing equality of means lies in comparing two different estimators of the common variance σ^2. Consider the pooled variance estimator:

$$MSE = \frac{1}{N-I}\sum_{i=1}^{I}\sum_{j=1}^{n_i}(Y_{ij} - \bar{Y}_{i\cdot})^2$$

where the subscript i denotes the group membership and the subscript j denotes the individual within group i; $N = \sum_{i=1}^{I} n_i$ where n_i is the i^{th} group's sample size. Clearly, the mean squared error (MSE, also called mean within group variability or MSW) is a pooled variance estimator, equal to the pooled sample variance S_p^2 when $I = 2$. It is straightforward to show MSE is unbiased for σ^2, given that the sample variance in each group is unbiased for the variance. Furthermore, since $\sum_{j=1}^{n_i}(Y_{ij} - \bar{Y}_{i\cdot})^2$ divided by σ^2 follows a Chi-squared distribution with $n_i - 1$ degrees of freedom, and since sums of independent Chi-squared r.v.'s are Chi-squared with df equal to the sum of degrees of freedom of the summand r.v.'s it follows that

$$\frac{MSE}{\sigma^2} \sim \chi^2(N-I).$$

Similarly, suppose the null hypothesis above is true and all the means are the same. Then, $\bar{Y}_{i\cdot} \overset{ind}{\sim} N(\mu, \sigma^2/n_i)$, and another unbiased estimator of σ^2 is given by

$$MSTr = \sum_{i=1}^{I} n_i(\bar{Y}_{i\cdot} - \bar{Y}_{\cdot\cdot})^2,$$

which is the mean square of treatments (or mean between group variation MSB). However, this is clearly only unbiased under H_0, because under H_a the observed

residuals inside the sum of squares are not centered. Indeed, under the alternative hypothesis, it is not hard (but is tedious) to show that

$$E(MSTr) = \sigma^2 + \frac{1}{I-1}\sum_{i=1}^{I} n_i(\mu_i - \bar{\mu})^2$$

where $\bar{\mu} = N^{-1}\sum_{i=1}^{I} n_i\mu_i$. The bias term is positive, which suggests a one-sided test of H_0 determined by the ratio $MSTr/MSE$. All we need is the sampling distribution of that statistic. For essentially the same reason as above, it follows that $MSTr$ follows a Chi-squared distribution with df $I-1$ under H_0. It's less obvious that MSTr and MSE are independent, but they are. Very briefly, the reason is that both of these statistics are *quadratic forms* that may be written $MSE = Y^\top AY$ and $MSTr = Y^\top BY$ for some matrices A and B. Multivariate normal distribution theory says that quadratic forms are independent if A and B are orthogonal matrices. Some careful work will show MSE and MSTr satisfy this property. Then, their ratio is a ratio of independent Chi-squared r.v.'s, which is an F r.v.:

$$F = \frac{MSTr}{MSE} \overset{H_0}{\sim} F(I-1, N-I).$$

Recall that only large values of this ratio suggest the null hypothesis is incorrect, and, therefore, we reject H_0 at level α if $F > F_{1-\alpha,I-1,N-I}$. This is called the analysis of variance (ANOVA) F test.

«««< HEAD ### Example: Donuts

Below we evaluate the F test for equality of mean oil absorption across all four types of oil. "By hand" we find a p-value of 0.0069, which suggests we should reject the null hypothesis of equal means and conclude different oils have different average absorption. Alternatively, the built-in `aov` function will compute the F statistic and p-value, and display both using the `summary` utility function.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
by_oil <- group_by(donuts, Oil)
means <- summarise(by_oil, means = mean(y))
```

```
SSE <- sum((donuts$y[donuts$Oil==1] - means$means[1])^2) + sum((donuts$y[donuts$Oil==2] - means$m
SSTr <- 6*((means$means[1] - mean(donuts$y))^2)+6*((means$means[2] - mean(donuts$y))^2)+6*((means
F <- (SSTr/3)/(SSE/(6*4-4))
SSE
```

```
## [1] 2018
```

```
SSTr
```

```
## [1] 1636.5
```

```
F
```

```
## [1] 5.406343
```

```
p.value <- 1-pf(F, 4-1, 6*4 - 4)
p.value
```

```
## [1] 0.006875948
```

```
my.aov <- aov(y~Oil, data=donuts)
summary(my.aov)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Oil          3   1636   545.5   5.406 0.00688 **
## Residuals   20   2018   100.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.3 Follow-up testing

When we reject the null hypothesis of equal means we usually want to know the answer to the follow-up question "which means are actually different"? It may be that one population mean is different from the others, which are all the same; or, it could be that every mean is different. If this question affects the actions the experimenters would take or recommend, then a follow-up test to evaluate these possibilities is valuable.

The most straightforward way to conduct such a follow-up test would be to compare all the means pairwise and use a Bonferroni correction to account for

the multiple tests. This is not the most powerful testing method. A much better method was developed by Tukey for use specifically with one-way ANOVA.

Tukey's procedure is based on a sampling distribution he derived called **Tukey's Studentized Range distribution**. If Z_i for $i = 1, \ldots, n$ are iid standard normal random variables and W is a Chi-Squared random variable with ν degrees of freedom independent from the Z_i's then the random variable

$$T = \frac{\max_{i \neq j} |Z_i - Z_j|}{\sqrt{W/\nu}}$$

follows Tukey's distribution with two degrees of freedom parameters, n and ν.

First, consider a balanced experiment, meaning we collect a random sample $Y_{i1,\ldots,Y_{i,n}}$ of size n from every population, $i = 1, \ldots, I$. We know that $n(I-1)MSE/\sigma^2$ has a Chi-squared distribution with $nI - I = n(I-1)$ degrees of freedom, and is independent from $\overline{Y}_{i\cdot}$ for $i = 1, \ldots, I$ (essentially by Student's Theorem). The variance of $\overline{Y}_{i\cdot} - \overline{Y}_{j\cdot}$ is $\sigma^2(1/n + 1/n)$. Therefore,

$$T = \frac{\max_{i \neq j} \left| \frac{\overline{Y}_{i\cdot}}{\sigma\sqrt{1/n}} - \frac{\overline{Y}_{j\cdot}}{\sigma\sqrt{1/n}} \right|}{\sqrt{n(I-1)MSE/\sigma^2}} = \frac{\max_{i \neq j} \left| \overline{Y}_{i\cdot} - \overline{Y}_{j\cdot} \right|}{\sqrt{MSE/n}}$$

follows Tukey's distribution with degrees of freedom I and $n(I-1)$.

Another way Tukey's test statistic is often written is

$$T = \frac{\max_{i \neq j} \left| \overline{Y}_{i\cdot} - \overline{Y}_{j\cdot} \right|}{2^{-1/2}\sqrt{MSE(2/n)}},$$

which may be easier to remember because the standard error of $\overline{Y}_{i\cdot} - \overline{Y}_{j\cdot}$ is $2\sigma^2/n$.

Tukey's procedure says to do the following: 1. Evaluate the test statistics $T_{i,j}$ for all $\binom{n}{2}$ pairs where

$$T_{i,j} = \frac{\left| \overline{Y}_{i\cdot} - \overline{Y}_{j\cdot} \right|}{2^{-1/2}\sqrt{MSE(2/n)}}.$$

2. Conclude μ_i and μ_j are significantly different at level α if $T_{i,j} > q_{1-\alpha,n,n(I-1)}$ where $q_{\alpha,u,v}$ is the α^{th} quantile of Tukey's distribution with u and v degrees of freedom. -OR- 1. Evaluate the $\binom{n}{2}$ confidence intervals

$$(\overline{Y}_{i\cdot} - \overline{Y}_{j\cdot} \pm 2^{-1/2} q_{1-\alpha,n,n(I-1)} \sqrt{MSE(2/n)}).$$

2. For every interval **not** including zero, conclude the corresponding means are significantly different.

Both procedures, the testing procedure and the confidence interval procedure, maintain Type 1 error at level α collectively over all $\binom{n}{2}$ pairwise comparisons.

When the experiment is **unbalanced**—meaning that the sample sizes of the samples taken from each of the I populations are not all equal, Tukey's procedure is approximate. In that case, Tukey's statistic

$$T_{i,j} = \frac{|\overline{Y}_{i\cdot} - \overline{Y}_{j\cdot}|}{2^{-1/2}\sqrt{MSE\left(\frac{1}{n_i} + \frac{1}{n_j}\right)}}$$

only approximately follows Tukey's studentized range distribution with degrees of freedom I and $N - I$ where $N = \sum_{i=1}^{I} n_i$. Tukey's procedure for unbalanced experiments is exactly the same, but the family-wise (over all comparisons) Type 1 error is only approximately controlled at level α.

6.3.1 Donuts example

Tukey's simultaneous pairwise comparisons may be performed in R using either the built-in TukeyHSD function or the emmeans and contrast functions in the emmeans package. The former performs the confidence interval procedure whereas the latter performs the test statistic procedure, but either way you will obtain the same results.

At level 5% only oils 2 and 4 had significantly different mean absorption. Note that the 4-2 CI does not contain zero, or, alternatively, that Oil2 - Oil4 p-value if 0.0039, the only one smaller than 0.05.

```
TukeyHSD(my.aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = y ~ Oil, data = donuts)
##
## $Oil
##     diff        lwr       upr     p adj
## 2-1   13  -3.232221 29.232221 0.1461929
## 3-1    4 -12.232221 20.232221 0.8998057
## 4-1  -10 -26.232221  6.232221 0.3378150
## 3-2   -9 -25.232221  7.232221 0.4270717
## 4-2  -23 -39.232221 -6.767779 0.0039064
## 4-3  -14 -30.232221  2.232221 0.1065573
```

```
library(emmeans)
fit.emm <- emmeans(my.aov, "Oil")
summary(contrast(fit.emm, method = "pairwise"))
```

```
##  contrast    estimate  SE df t.ratio p.value
##  Oil1 - Oil2      -13 5.8 20  -2.242  0.1462
##  Oil1 - Oil3       -4 5.8 20  -0.690  0.8998
##  Oil1 - Oil4       10 5.8 20   1.724  0.3378
##  Oil2 - Oil3        9 5.8 20   1.552  0.4271
##  Oil2 - Oil4       23 5.8 20   3.966  0.0039
##  Oil3 - Oil4       14 5.8 20   2.414  0.1066
##
## P value adjustment: tukey method for comparing a family of 4 estimates
```

6.4 Inference for particular contrasts

Besides pairwise comparisons, experimenters may be interested in comparing particular combinations of treatment means. For example, they may want to answer the question: "Is Oil 4 significantly different than Oils 1-3 combined?" in the contest of the donuts experiment. This is not a pairwise comparison; rather, it's a linear combination (or contrast) defined by $\mu_4 - \frac{1}{3}(\mu_1 + \mu_2 + \mu_3)$. It is important that we decide which contrasts to test *apriori*—that is, before we do the experiment. We do not want to bias our test procedures by using the data to suggest which contrasts might be significant and then testing those in particular. That practice leads to inflated Type 1 error. Likewise, when testing multiple contrasts, it is a good idea to make a Type 1 correction for multiple tests if a decision or action hinges on the outcome of the tests in total. For this correction we need something new, because Tukey's procedure only works for pairwise comparisons, and contrasts are more general objects.

In general, a contrast may be written $\gamma = \sum_{i=1}^{I} c_i \mu_i = c^\top \mu$ where $0 = \sum_{i=1}^{I} c_i$. The estimated contrast is $\hat{\gamma} = \sum_{i=1}^{n} c_i \overline{Y}_{i\cdot}$ and it has standard error $\sqrt{\sigma^2 \sum_{i=1}^{n} (c_i^2/n_i)}$. It follows that tests and CIs for γ are based on the follow Student's t r.v.:

$$\frac{\hat{\gamma} - \gamma}{\sqrt{\sigma^2 \sum_{i=1}^{n} (c_i^2/n_i)}} \sim t_{N-I}.$$

Scheff'e developed a correction for simultaneous inference on several (even all possible) contrasts. This method is prefereable to Bonferroni for more than a few contrasts, but will be overly conservative for just two or three contrasts. Scheff'e says to use cutoffs of the form $\sqrt{(I-1)F_{1-\alpha,I-1,N-I}}$ rather than Student's t

quantiles $t_{1-\alpha/2,N-I}$ for tests of contrasts and for margin of error expressions for CIs of contrasts. In particular, Scheff'e's CIs for contrasts γ_ℓ, $\ell = 1, ...$ are

$$\left(\hat{\gamma}_\ell \pm \sqrt{(I-1)F_{1-\alpha,I-1,N-I}}\sqrt{MSE\sum_{i=1}^{n}(c_i^2/n_i)}\right).$$

This collection of CIs has coverage $100(1-\alpha)\%$ for any vector of contrasts $\gamma = (\gamma_1, \gamma_2, ...)$.

6.4.1 Donuts Example

Below we compute CIs for the contrast comparing Oil 4 to the average of Oils 1-3 and p-values for the test of $\gamma = 0$ using both the single contrast methods based on the Student's t sampling distribution and Scheffe's correction for inference on all simultaneous contrasts. Using the emmeans package we can do these computations with built-in functions rather than "by hand". Note that the p-value using Scheff'e's method is about ten times higher than the single contrast method, which demonstrates the correction for inference on all contrasts is substantial.

```
MSE <- SSE/(6*4-4)
c1 <- c(-1/3, -1/3, -1/3, 1)
gamma.hat <- sum(c1*means$means)
se <- sqrt(MSE*sum((c1^2)*(1/6)))
c(gamma.hat - qt(0.975,4*6-4)*se,gamma.hat + qt(0.975,4*6-4)*se)
```

```
## [1] -25.54414  -5.78919
```

```
2*pt(gamma.hat/se, 4*6-4)
```

```
## [1] 0.003506899
```

```
c(gamma.hat - sqrt((4-1)*qf(0.95,4-1,4*6-4))*se,gamma.hat + sqrt((4-1)*qf(0.95,4-1,4*6-4))*se)
```

```
## [1] -30.103372  -1.229962
```

```
1-pf((gamma.hat/(se*sqrt(4-1)))^2, 4-1,4*6-4)
```

```
## [1] 0.03014771
```

```
library(emmeans) # may need to install.packages('emmeans')
fit.emm <- emmeans(my.aov, "Oil")
# unadjusted, just testing one contrast
summary(contrast(fit.emm, method = list(c(-1/3, -1/3, -1/3, 1))))
```

```
##  contrast                                                          estimate   SE
##  c(-0.333333333333333, -0.333333333333333, -0.333333333333333,     -15.7 4.74
##  df t.ratio p.value
##  20  -3.309  0.0035
```

```
# Scheffe adjustment so that we could test all linear contrasts
summary(contrast(fit.emm, method = list(c(-1/3, -1/3, -1/3, 1)),
                 adjust = "scheffe"), scheffe.rank = 3)
```

```
##  contrast                                                          estimate   SE
##  c(-0.333333333333333, -0.333333333333333, -0.333333333333333,     -15.7 4.74
##  df t.ratio p.value
##  20  -3.309  0.0301
##
## P value adjustment: scheffe method with rank 3
```

6.5 Checking Assumptions in one-way ANOVA

The ANOVA F test requires the residuals are normally distributed with homogeneous (equal/same) variance. The F test also requires independence, but we will not consider checking that assumption.

To check for equal variances we may apply Levene's test, which is essentially a generalization of the two-sample F test for equality of two normal population variances. To apply Levene's test we compute the residuals $e_{ij} = Y_{ij} - \overline{Y}_{i\cdot}$ and perform the ANOVA F test on the absolute residuals $|e_{ij}|$. The null and alternative hypotheses of Levene's test (ANOVA F test on residuals) are H_0 : $\sigma_i^2 = \sigma^2$, for all $i = 1, \ldots, I$ versus H_a : at least one population variance is different than the other $I - 1$ variances.

We may check for normality visually using qq-plots of (studentized) residuals. If the assumption is true, we expect almost all the points to fall between $(-2, 2)$ and close to the diagonal. There are formal tests for normality (Kolmogorov-Smirnov, Anderson-Darling, etc.) but these are not very powerful.

If qq-plots suggest the normality assumption may not be satisfied, then we can apply a non-parametric test to evaluate equality of the several population distributions. Just the like rank-sum test for comparing two populations, a procedure based on ranks can be used to compare I populations. Let r_{ij} be the

ranks of the y_{ij} and let $\bar{r}_{i\cdot}$ and $\bar{r}_{\cdot\cdot}$ be the averages of ranks in each population sample and overall, respectively. Then, the Kruskal-Wallis test statistic is the ratio of the SSB (or called SSTr) to SST of the ranks, multiplied by $N-1$:

$$\chi^2 = (N-1) \cdot \frac{\sum_{i=1}^{I} n_i(\bar{r}_{i\cdot} - \bar{r}_{\cdot\cdot})^2}{\sum_{i=1}^{I}\sum_{j=1}^{n_i}(r_{ij} - \bar{r}_{\cdot\cdot})^2}.$$

The null hypothesis of the Kruskall-Wallis test is that the I populations are really all the same population (the I distributions are equal). In the special case in which we assume all populations are normal with the same mean, this null hypothesis is equivalent to the assertion all the means are equal. Under the Kruskal-Wallis null hypothesis, the test statistic χ^2 is *approximately* distributed as a Chi-squared r.v. with $I-1$ degrees of freedom.

6.5.1 Example: Cehcking assumptions for donuts experiment

We can use the residuals computed by the aov function when we performed the ANOVA F test on the responses to perform Levene's test (the ANOVA F test on the residuals). alternatively, we can use the built-in functions for Bartlett's test or Levene's test in the car package. Bartlett's test is similar to Levene's test, but is said to be more sensitive to non-normality of responses. Sometimes Leven's test is performed with sample medians replacing sample means, and is referred to as the Brown-Forsythe test. That test is even less sensitive to departures from the normality assumption.

For the donuts data we do not reject the hypothesis of equal variances—our assumption seems reasonable.

```
donuts$residuals <- abs(my.aov$residuals)
my.aov.resid <- aov(residuals~Oil, data = donuts)
summary(my.aov.resid)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Oil          3   41.8   13.94   0.361  0.782
## Residuals   20  772.0   38.60
```

In R you can perform the Brow-Forsythe test using the leveneTest function in teh car package with option center = "median"; by default, the function performs Levene's test.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
bartlett.test(y ~ Oil, data = donuts)
```

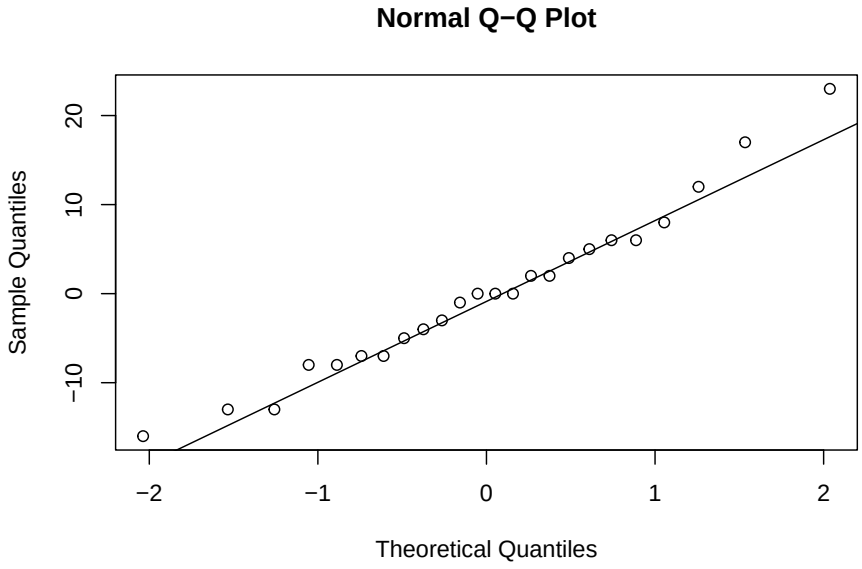
```
##
##  Bartlett test of homogeneity of variances
##
## data:  y by Oil
## Bartlett's K-squared = 1.7504, df = 3, p-value = 0.6258
```

```
leveneTest(y ~ Oil, data = donuts)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  3  0.3434 0.7942
##       20
```

Below we perform the Kruskal-Wallis test "by hand" and using its built-in function. For the donuts data, there is no reason to do this, however, because the qq-plot suggests normality is reasonable.

```
qqnorm(my.aov$residuals)
qqline(my.aov$residuals)
```




```
rij <- rank(donuts$y)

r1 <- mean(rij[donuts$Oil == 1])
r2 <- mean(rij[donuts$Oil== 2])
r3 <- mean(rij[donuts$Oil == 3])
r4 <- mean(rij[donuts$Oil == 4])

# SSE

SSE <- sum((rij[donuts$Oil == 1] - r1)^2)+sum((rij[donuts$Oil == 2] - r2)^2)+sum((rij[donuts$Oil
SSE
```

```
## [1] 557.4167
```

```
SST <- sum((rij - mean(rij))^2)
SST
```

```
## [1] 1148
```

```
# SSTr
SSTr <- SST-SSE
SSTr
```

```
## [1] 590.5833
```

```
test.stat <- (6*4 - 1)*(SSTr / SST)
test.stat
```

```
## [1] 11.83224
```

```
1-pchisq(test.stat, 4-1)
```

```
## [1] 0.007980479
```

```
kruskal.test(y ~ Oil, data = donuts)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  y by Oil
## Kruskal-Wallis chi-squared = 11.832, df = 3, p-value = 0.00798
```

Chapter 7

Randomized Complete Block Design

In this chapter we discuss the technique of *blocking* which means to sample individuals from groups that are known (or strongly suspected) to have different average responses. Blocking is a type of explicit control—we account for the known effect of the blocks on the response through our sampling/experimental design. The effect of including blocks is (usually) to reduce the remaining variability in the response, thereby making it easier to detect treatment effects. One way to visualize this blocking effect is to recall the sum of squares decomposition SST = SSE + SSTr. When blocks are utilized, the decomposition becomes SST = SSE + SSBl + SSTr. The block sum of squares tends to absorb mostly the SSE term, thereby reducing the SSE and making the F ratio for treatments larger. However, some of the information of the data is "used up" estimating the block effects, and this reduces the degrees of freedom associated with MSE. Therefore, if "useless" blocks are used, we will tend to lose efficiency/power in detecting treatment effects.

7.1 Paired experiments as blocking

We are already familiar with blocking in a special type of two-sample experiment—paired data. For example, consider a paired design where responses on n individuals are measured as X_i "before" and Y_i "after" a treatment. The individuals constitute n blocks, with one sample, the difference $D_i = Y_i - X_i$, for each block. The variance of D_i is $V(D_i) = V(X_i) + V(Y_i) - 2Cov(X_i, Y_i)$. In a two-sample experiment where two different groups of individuals are treated, the covariance of responses between groups is zero. However, if multiple measurements are taken on the

same individual, we generally expect the covariance of those measurements to be positive. Therefore, intuitively, we would expect the variance of differences of observations on the same individual to be lower than the variance of the difference of responses on two different individuals.

7.2 Randomized Complete Block Designs

7.2.1 Notation

We will use the following notation: - n is the number of blocks - J is the number of treatments - r is the number of replications of each treatment in a block. So, each block has rJ responses. - Y_{ijk} is the response for the i^{th} block for treatment j at the k^{th} replicate. - $\overline{Y}_{i\cdot\cdot} = (rJ)^{-1}\sum_{j=1}^{J}\sum_{k=1}^{r} Y_{ijk}$ is the sample mean response of i^{th} block - $\overline{Y}_{\cdot j\cdot} = (rn)^{-1}\sum_{i=1}^{n}\sum_{k=1}^{r} Y_{ijk}$ is the sample mean response of j^{th} treatment - $\overline{Y}_{\cdots} = (rJn)^{-1}\sum_{i=1}^{n}\sum_{j=1}^{J}\sum_{k=1}^{r} Y_{ijk}$ is the sample mean response of all the responses, the grand sample mean. In single replication studies, $r = 1$ and the third subscript is dropped from the notation.

The model may be written in the following effects-style notation:

$$Y_{ijk} = \mu + \beta_i + \tau_j + \epsilon_{ijk}$$

where μ is an "intercept" term equal to $E(Y_{ijk})$ if the other effects are actually zero; β_i for $i = 1, \ldots, n$ are the block effects, τ_j for $j = 1, \ldots, J$ are the treatment effects, and ϵ_{ijk} is the effect of random sampling variability where $\epsilon_{ijk} \overset{iid}{\sim} N(0, \sigma^2)$.

The above model requires two constraints in order to estimate the parameters (μ, β_i, τ_j) for all i and j. Here is the problem: the estimates of these parameters will be functions of the $1 + n + J$ sample means above (the grand sample mean, block and treatment sample means). But, these are not all freely varying. For instance, since $\overline{Y}_{\cdots} = (J)^{-1}\sum_{j=1}^{J}\overline{Y}_{\cdot j\cdot}$, it follows that we only need to know $J-1$ of the J treatment sample means $\overline{Y}_{\cdot j\cdot}$ and $\overline{Y}_{\cdots}$ in order to determine the last one. The same is true of the block sample means. This means we must introduce two constraints on the parameters (μ, β_i, τ_j) for all i and j to avoid defining parameters that are simply linear combinations of other parameters.

There are two common choices, the baseline constraints and the sum-to-zero constraints. Under the baseline constraints we set one of the treatment effects and one of the block effects to zero. Often, we would set $\beta_n = 0$ and $\tau_J = 0$, but sometimes (in some software) the first rather than last effects are set to zero. Whichever are set to zero are absorbed by μ and μ becomes the "baseline" effect. The remaining non-zero effects represent departures from the baseline. The

second common constraint is the sum-to-zero constraint, which forces $\sum \beta_i = \sum \tau_j = 0$. This constraint causes μ to become the overall average response, and the effects to represent departures from the average.

7.2.2 Example: Penicillin Manufacturing

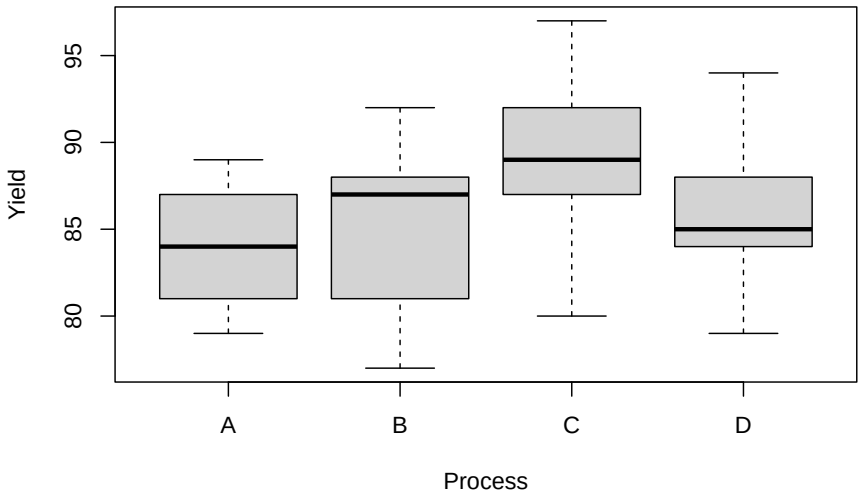
In this example 5 batches of raw organic material are used to make the antibiotic penicillin. We are interested in the yield of penicillin using 4 different industrial processes. Each process is used once per batch.

```
penicillin <- read.csv('penicillin.csv')
head(penicillin)
```

```
##   Batch Process Yield
## 1     1       A    89
## 2     1       B    88
## 3     1       C    97
## 4     1       D    94
## 5     2       A    84
## 6     2       B    77
```

```
penicillin$Batch <- factor(penicillin$Batch, levels = unique(penicillin$Batch))
penicillin$Process <- factor(penicillin$Process, levels = unique(penicillin$Process))
```

```
boxplot(Yield ~ Process,
        data = penicillin)
```



Let's see what parameter estimates R produces for the penicillin data. From the output of the lm function we see that treatment A and Batch 1 are missing—these are the baseline and are reflected in the intercept estimate, which is 90. How did R get 90? Well, the overall sample mean is 86; the process A mean yield is 84, and the batch 1 mean yield is 92. So, the effect of process A compared to the overall average is -2 and the effect of batch 1 compared to the overall average is +8. This combines for +6, so we end up with a batch1+processA baseline effect of 90 = 84+6. The rest of the effect estimates are easy to obtain as differences from this baseline. For example, since process B has a sample mean 1 larger than A, the process B effect is 1. And, since batch 3 has a sample mean effect 7 less than batch 1 the batch 3 effect is -7, and so on.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
by_batch <- group_by(penicillin, Batch)
by_batch <- summarise(by_batch, batch_mean = mean(Yield))
by_batch
```

```
## # A tibble: 5 x 2
##   Batch batch_mean
##   <fct>      <dbl>
## 1 1             92
## 2 2             83
## 3 3             85
## 4 4             88
## 5 5             82
```

```
by_trt <- group_by(penicillin, Process)
by_trt <- summarise(by_trt, trt_mean = mean(Yield))
by_trt
```

```
## # A tibble: 4 x 2
##   Process trt_mean
##   <fct>      <dbl>
## 1 A             84
## 2 B             85
## 3 C             89
## 4 D             86
```

```
mean(penicillin$Yield)
```

```
## [1] 86
```

```
my.lm <- lm(Yield~Process+Batch, data = penicillin)
summary(my.lm)
```

```
##
## Call:
## lm(formula = Yield ~ Process + Batch, data = penicillin)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -5.00  -2.25  -0.50   2.25   6.00
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   90.000      2.745  32.791  4.1e-13 ***
## ProcessB       1.000      2.745   0.364  0.72194
## ProcessC       5.000      2.745   1.822  0.09351 .
## ProcessD       2.000      2.745   0.729  0.48018
## Batch2        -9.000      3.069  -2.933  0.01254 *
## Batch3        -7.000      3.069  -2.281  0.04159 *
## Batch4        -4.000      3.069  -1.304  0.21686
## Batch5       -10.000      3.069  -3.259  0.00684 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.34 on 12 degrees of freedom
## Multiple R-squared:  0.5964, Adjusted R-squared:  0.361
## F-statistic: 2.534 on 7 and 12 DF,  p-value: 0.07535
```

The *options* statement below causes R to use the sum-to-zero constraints. We can see the resulting parameter estimates are now in terms of differences from the overall sample mean yield.

```
options(contrasts=c('contr.sum', 'contr.sum'))
my.lm.sum <- lm(Yield~Process+Batch, data = penicillin)
summary(my.lm.sum)
```

```
##
## Call:
## lm(formula = Yield ~ Process + Batch, data = penicillin)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -5.00  -2.25  -0.50   2.25   6.00
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  86.0000     0.9704  88.624  < 2e-16 ***
## Process1     -2.0000     1.6808  -1.190  0.25708
## Process2     -1.0000     1.6808  -0.595  0.56292
## Process3      3.0000     1.6808   1.785  0.09956 .
## Batch1        6.0000     1.9408   3.092  0.00934 **
## Batch2       -3.0000     1.9408  -1.546  0.14812
## Batch3       -1.0000     1.9408  -0.515  0.61573
## Batch4        2.0000     1.9408   1.031  0.32310
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.34 on 12 degrees of freedom
```

```
## Multiple R-squared:  0.5964, Adjusted R-squared:  0.361
## F-statistic: 2.534 on 7 and 12 DF,  p-value: 0.07535
```

We can obtain design matrices under both sets of constraints using the *model.matrix* function. Note that the sum-to-zero design matrix literally has columns summing to zero!

```
X <- model.matrix(my.lm)
X
```

```
##    (Intercept) ProcessB ProcessC ProcessD Batch2 Batch3 Batch4 Batch5
## 1            1        0        0        0      0      0      0      0
## 2            1        1        0        0      0      0      0      0
## 3            1        0        1        0      0      0      0      0
## 4            1        0        0        1      0      0      0      0
## 5            1        0        0        0      1      0      0      0
## 6            1        1        0        0      1      0      0      0
## 7            1        0        1        0      1      0      0      0
## 8            1        0        0        1      1      0      0      0
## 9            1        0        0        0      0      1      0      0
## 10           1        1        0        0      0      1      0      0
## 11           1        0        1        0      0      1      0      0
## 12           1        0        0        1      0      1      0      0
## 13           1        0        0        0      0      0      1      0
## 14           1        1        0        0      0      0      1      0
## 15           1        0        1        0      0      0      1      0
## 16           1        0        0        1      0      0      1      0
## 17           1        0        0        0      0      0      0      1
## 18           1        1        0        0      0      0      0      1
## 19           1        0        1        0      0      0      0      1
## 20           1        0        0        1      0      0      0      1
## attr(,"assign")
## [1] 0 1 1 1 2 2 2 2
## attr(,"contrasts")
## attr(,"contrasts")$Process
## [1] "contr.treatment"
##
## attr(,"contrasts")$Batch
## [1] "contr.treatment"
```

```
X <- model.matrix(my.lm.sum)
X
```

```
##    (Intercept) Process1 Process2 Process3 Batch1 Batch2 Batch3 Batch4
```

```
## 1            1        1        0        0      1      0      0      0
## 2            1        0        1        0      1      0      0      0
## 3            1        0        0        1      1      0      0      0
## 4            1       -1       -1       -1      1      0      0      0
## 5            1        1        0        0      0      1      0      0
## 6            1        0        1        0      0      1      0      0
## 7            1        0        0        1      0      1      0      0
## 8            1       -1       -1       -1      0      1      0      0
## 9            1        1        0        0      0      0      1      0
## 10           1        0        1        0      0      0      1      0
## 11           1        0        0        1      0      0      1      0
## 12           1       -1       -1       -1      0      0      1      0
## 13           1        1        0        0      0      0      0      1
## 14           1        0        1        0      0      0      0      1
## 15           1        0        0        1      0      0      0      1
## 16           1       -1       -1       -1      0      0      0      1
## 17           1        1        0        0     -1     -1     -1     -1
## 18           1        0        1        0     -1     -1     -1     -1
## 19           1        0        0        1     -1     -1     -1     -1
## 20           1       -1       -1       -1     -1     -1     -1     -1
## attr(,"assign")
## [1] 0 1 1 1 2 2 2 2
## attr(,"contrasts")
## attr(,"contrasts")$Process
## [1] "contr.sum"
##
## attr(,"contrasts")$Batch
## [1] "contr.sum"
```

7.2.3 Sums of squares and F tests in RCBD

The primary goal of RCBD experiments is usually to evaluate whether all the treatment mean responses are equal, just as in one-way ANOVA. And, as in *completely randomized experiments* — experiments in which a random sample of experimental units are randomly assigned treatments — an F test based on the ratio of two variance estimators is useful for testing this hypothesis. Keep in mind that we are not interested in testing whether the block mean responses are all equal, although we will consider a measure of the usefulness of blocking later on.

Just as in the completely randomized design, for RCBD experiments we can decompose the total sum of squares into components for blocks, treatments, and the "errors":

$$SST = SSE + SSBl + SSTr$$

$$SST = \sum_{i=1}^{n}\sum_{j=1}^{J}\sum_{k=1}^{r}(y_{ijk} - \overline{y}_{...})^2$$

$$SSBl = (rJ)\sum_{i=1}^{n}(\overline{y}_{i..} - \overline{y}_{...})^2$$

$$SSTr = (rn)\sum_{j=1}^{J}(\overline{y}_{.j.} - \overline{y}_{...})^2$$

$$SSE = \sum_{i=1}^{n}\sum_{j=1}^{J}\sum_{k=1}^{r}(y_{ijk} - \overline{y}_{i..} - \overline{y}_{.j.} + \overline{y}_{...})^2$$

To test the hypothesis $H_0 : \tau_j = 0$, $j = 1, \dots, J$, or equivalently, that the treatments have no effect (the mean response is constant over treatments) versus H_a : not all treatment effects are 0 , i.e., $H_a : \tau_j \neq 0$, for some $j \in \{1, \dots, J\}$, we use the following F statistic:

$$F = \frac{SSTr/(J-1)}{SSE/(nJr - n - J + 1)} \overset{H_0}{\sim} F_{J-1, nJr-n-J+1}.$$

To compute degrees of freedom of a sum of squares, the general rule is to subtract from the total number of summands the number of estimated parameters. For example, in SSTr there are J summands and 1 "estimated parameter", which is the parameter estimated by $\overline{y}_{...}$. And, in SSE there are nJr summands, and there are $n + J + 1 - 2$ uniquely determined sample means in the set $\{\overline{y}_{1..}, \dots, \overline{y}_{n..}, \overline{y}_{.1.}, \dots, \overline{y}_{.J.}, \overline{y}_{...}\}$ as discussed above.

7.2.4 Example: Significance of Process in Penicillin experiment

The F test does not suggest mean yield varies over the four penicillin manufacturing processes.

```
SST <- sum((penicillin$Yield - mean(penicillin$Yield))^2)
SSTr <- 5*sum((by_trt$trt_mean - mean(penicillin$Yield))^2)
SSBl <- 4*sum((by_batch$batch_mean - mean(penicillin$Yield))^2)
SSE <- SST - SSTr - SSBl
SSTr
```

```
## [1] 70
```

```
SSBl
```

```
## [1] 264
```

```
SSE
```

```
## [1] 226
```

```
F <- (SSTr/(4-1))/(SSE / (5*4-4-5+1))
F
```

```
## [1] 1.238938
```

```
1-pf(F,4-1,5*4-4-5+1)
```

```
## [1] 0.3386581
```

```
my.aov <- aov(Yield~Batch+Process, penicillin)
my.aov
```

```
## Call:
##    aov(formula = Yield ~ Batch + Process, data = penicillin)
##
## Terms:
##                   Batch Process Residuals
## Sum of Squares      264      70       226
## Deg. of Freedom       4       3        12
##
## Residual standard error: 4.339739
## Estimated effects may be unbalanced
```

```
summary(my.aov)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Batch        4    264   66.00   3.504 0.0407 *
## Process      3     70   23.33   1.239 0.3387
## Residuals   12    226   18.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

7.2.5 Follow-up comparisons and contrasts

Just as in one-way ANOVA for a completely randomized design (CRD) we may analyze pairwise comparisons of treatment mean responses when the F test rejects the hypothesis that all treatment means are the same. Or, we could estimate and test planned contrasts. And, there is essentially no differnce in how we apply these estimates and tests compared with CRD.

To compute Tukey HSD confidence intervals we use essentially the same formula:

$$\left(\overline{Y}_{i\cdot\cdot} - \overline{Y}_{\ell\cdot\cdot} \pm 2^{-1/2} q_{1-\alpha,\, J,\, nJr-n-J+1} \sqrt{MSE\left(\frac{1}{rn} + \frac{1}{rn}\right)}\right)$$

to define a set of simultaneous $100(1-\alpha)\%$ CIs for the pairwise differences of treatment mean responses. (Recall $q_{\alpha,\,u,\,v}$ is the α quantile of Tukey's Studentized range distribution for u means and with v df.)

Let $\gamma = c^\top \mu = \sum_{i=1}^J c_i \mu_i$ denote a linear combination of treatment mean responses where $\sum_{i=1}^J c_i = 0$. Then, γ is a contrast, and

$$t = \frac{\hat{\gamma} - \gamma}{\sqrt{MSE \sum_{i=1}^J \left(\frac{c_i^2}{nr}\right)}} \sim t_{nrJ-n-J+1},$$

where $\hat{\gamma} = \sum_{i=1}^J c_i \overline{Y}_{i\cdot\cdot}$. This t random variable may be used to derive point null hypothesis tests and CIs for γ.

For simultaneous tests or CIs for a set of contrasts, we may use Scheff'e's method. The Scheff'e simultaneous $100(1-\alpha)\%$ CIs for all linear contrasts are given by:

$$\left(\hat{\gamma} \pm \sqrt{(J-1) F_{1-\alpha,J-1,nJr-n-J+1}} \sqrt{MSE \sum_{i=1}^J \left(\frac{c_i^2}{nr}\right)}\right).$$

7.2.6 Example: Tukey, contrasts, and Scheffe for penicillin experiment

There is no reason to use Tukey's method with the penicillin experiment given the F test supports the null hypothesis. But, just for practice we can compute the Process B - Process A CI and compare with the built-in TukeyHSD function.

```
c(by_trt$trt_mean[2] - by_trt$trt_mean[1] - sqrt(.5)*qtukey(0.95, 4, 4*5-4-5+1)*sqrt((SSE / (5*4-
by_trt$trt_mean[2] - by_trt$trt_mean[1] + sqrt(.5)*qtukey(0.95, 4, 4*5-4-5+1)*sqrt((SSE / (5*4-4-
```

```
## [1] -7.148719  9.148719
```

```
TukeyHSD(my.aov, which = 'Process')
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Yield ~ Batch + Process, data = penicillin)
##
## $Process
##     diff        lwr       upr     p adj
## B-A    1  -7.148719  9.148719 0.9826684
## C-A    5  -3.148719 13.148719 0.3105094
## D-A    2  -6.148719 10.148719 0.8837551
## C-B    4  -4.148719 12.148719 0.4905194
## D-B    1  -7.148719  9.148719 0.9826684
## D-C   -3 -11.148719  5.148719 0.7002271
```

Next, let's test is the Process C mean yield is different than the average of the other processes. This is a contrast with $c = (-1/3, -1/3, 1, -1/3)$. We might be tempted to test this particular contrast after peeking at the boxplot above. That's not a good idea! Using the data to suggest tests to perform leads to inflated Type 1 errors. However, if we apply Scheff'e's correction, then we control the Type 1 error rate for all linear contrasts, which includes comparisons that are *inspired* by the data.

At $\alpha = .10$ we would actually reject the null hypothesis that the Process C mean is equal to the average of the A, B, and D means if we performed the uncorrected test (which we should not because we chose to based on the boxplot). But, the Scheff'e correction saves us from our shameful p-value snooping!

```
hat.gamma <- by_trt$trt_mean[3] - mean(c(by_trt$trt_mean[c(1,2,4)]))
t <- hat.gamma / sqrt((SSE / (5*4-4-5+1)) * (1/5 + 3*(1/(5*9))))
2*(1-pt(abs(t), 5*4-4-5+1)) # uncorrected p-value
```

```
## [1] 0.09955857
```

```
F <- (t/sqrt(4-1))^2
1-pf(F, 4-1, 5*4-4-5+1) # Scheffe corrected p-value
```

```
## [1] 0.4014427
```

```
library(emmeans)
fit.emm <- emmeans(my.aov, "Process")
# unadjusted, just testing one contrast
summary(contrast(fit.emm, method = list(c(-1/3, -1/3, 1,-1/3))))
```

```
##  contrast                                                          estimate   SE
##  c(-0.333333333333333, -0.333333333333333, 1, -0.333333333333333        4 2.24
##  df t.ratio p.value
##  12   1.785  0.0996
##
## Results are averaged over the levels of: Batch
```

```
# Scheffe adjustment so that we could test all linear contrasts
summary(contrast(fit.emm, method = list(c(-1/3, -1/3, 1,-1/3)),
                 adjust = "scheffe"), scheffe.rank = 3)
```

```
##  contrast                                                          estimate   SE
##  c(-0.333333333333333, -0.333333333333333, 1, -0.333333333333333        4 2.24
##  df t.ratio p.value
##  12   1.785  0.4014
##
## Results are averaged over the levels of: Batch
## P value adjustment: scheffe method with rank 3
```

7.2.7 Relative efficiency of Blocking (vs. CRD)

After an RCBD experiment is performed it is sometimes helpful for researchers to understand whether blocking was a helpful part of the experimental design. Blocking variables are selected by the researchers with the belief their levels are associated with different average responses. The idea is that if we take into account the variability in response due to block variable levels, we will reduce the unexplained variability. This manifests in the decomposition

$$SST = SSE + SSBl + SSTr$$

where we expect the block sum of squares to "absorb" part of the SSE that would have been present in a CRD. Smaller SSE in turn increases the F statistic $F = MSTr/MSE$ for testing significance of the treatments by lowering the denominator in the ratio. On the other hand, blocking creates an opposite effect that decreases this F statistic as well. Just like the total sum of squares decomposes, so do the degrees of freedom:

$$nJr - 1 = (nJr - n - J + 1) + (n - 1) + (J - 1).$$

The upshot is that the number of degrees of freedom "used" in estimating SSE is smaller if blocks are included compared to a CRD with no blocks and equal total sample size. This reduces the denominator of MSE, which increases MSE relative to CRD and thus decreases the F ratio for testing for significant treatment effects. Usually, if experimenters choose a small number of intuitively useful blocks, RCBD is "more efficient" than CRD in the sense that its F test has increased power to detect treatment effects.

We can attempt to quantify the effect of blocking compared to CRD using a post-experiment measure called the relative efficiency (RE) of blocking (relative, that is, to CRD). The formula, given below, accounts for both the decrease in degrees of freedom and the approximate decrease in $\hat{\sigma}^2$ when blocking (RCBD) versus not blocking (CRD):

$$RE = \frac{df_{MSE,RCBD}+1}{df_{MSE,CRD}+1}\frac{df_{MSE,CRD}+3}{df_{MSE,RCBD}+3}\frac{\hat{\sigma}^2_{CRD}}{\hat{\sigma}^2_{RCBD}},$$

where

$$\hat{\sigma}^2_{CRD} = MSE_{CRD},$$

$$MSE_{CRD} = \frac{df_{Bl}MSBl_{RCBD} + (df_{Tr}+df_E)MSE_{RCBD}}{df_{Bl}+df_{Tr}+df_E}.$$

And, from above, we have

$df_{Tr} = J-1$, $df_{Bl} = n-1$, $df_E = df_{MSE,RCBD} = nJr-n-J+1$, and $df_{MSE,CRD} = nJr-J$.

7.2.8 Example: RE of blocking in penicillin experiment

The relative efficiency of blocking is about 1.48, which can be interpreted as saying each replicated response in the RCBD experiment provides approximately 1.48 times at much (48% more) information compared to a replicate in a CRD design. In other words, it would take 48% more samples in a CRD to get the same amount of information as in a RCBD. As long as RE is above 1, the RCBD design provides an advantage over CRD. Since it may be substantially more work to implement a RCBD than a CRD, experimenters would like to see a substantial advantage in RE for RCBD relative to CRD.

```
MSE_RCBD <- SSE / (4*5-4-5+1)
MSBl <- SSBl / (5-1)
MSE_CRD <- ((5-1)*MSBl + (5*4-5-4+1 + 4-1)*MSE_RCBD)/(5*4-5-4+1+4-1+5-1)
df_MSE_CRD <- 5*4-4
df_MSE_RCBD <- 5*4-4-5+1

RE <- (MSE_CRD/MSE_RCBD) * ((df_MSE_CRD+3)/(df_MSE_RCBD +3)) * ((df_MSE_RCBD+1)/(df_MSE
RE
```

```
## [1] 1.479334
```

Chapter 8

Latin Squares for two blocking variables

8.1 Designs for multiple blocks

Suppose we have two blocking variables and one treatment variable, and suppose all three of these categorical variables have the same number of levels. For example, suppose an experiment to measure the miles per gallon (MPG) of automobiles using 4 types of gasoline fuel additives (the treatments) is blocked by 4 models of car and 4 different drivers driving the cars on the same road course. In order to carry out an RCBD we would consider the 16 crossed blocks given by driver times car model. And, we require each treatment to be replicated at least once per block, which means we need a total of at least 64 samples.

The Latin square design is a special design for the case of 2 (or more) blocking variables and one treatment variable all having the same number of levels. The Latin square (or cube, or hypercube!) only requires one treatment be used for each combination of blocks, but according to a strict pattern: If we write down a grid or matrix corresponding to crossing the levels of the two blocking variables, then a Latin square design corresponds to every treatment level appearing in each row and column exactly once (think of a sudoku puzzle).

8.1.1 Example: MPG experiment

Four for drivers (1,2,3,4), four cars (1,2,3,4), and four treatments/additives (A,B,C,D) we have the following Latin square design:

		Driver			
		1	2	3	4
Car	1	A	B	C	D
	2	B	C	D	A
	3	C	D	A	B
	4	D	A	B	C

8.2 Model notation

A Latin square design may be represented by the follwowing model:

$$Y_{ijk} = \mu + \beta_i + \gamma_j + \tau_k + \epsilon_{ijk}$$

where $i, j, k = 1, \ldots, r$, β and γ refer to row and column block effects and τ is the treatment effect. As always, the model is over-parametrized, and constraints are necessary; for example, the baseline constraint $\beta_r = \gamma_r = \tau_r = 0$.

8.3 Sums of squares and test for treatment effects

The row (SSBr) and column (SSBc) block sums of squares, and treatment (SSTr) sums of squares are given by:

$$SSBr = r\sum_i (\overline{Y}_{i\cdot\cdot} - \overline{Y}_{\cdots})^2, \quad SSBc = r\sum_j (\overline{Y}_{\cdot j\cdot} - \overline{Y}_{\cdots})^2, \quad SSBr = r\sum_k (\overline{Y}_{\cdot\cdot k} - \overline{Y}_{\cdots})^2.$$

Each have $r-1$ degrees of freedom (r summands minus 1 estimated parameter, which is $\overline{Y}_{\cdots}$ estimating the overall mean parameter). The total sum of squares is

$$SST = \sum_i \sum_j \sum_k (Y_{ijk} - \overline{Y}_{\cdots})^2$$

which has $r^2 - 1$ degrees of freedom (r^2 summands minus one estimated parameter, which is $\overline{Y}_{\cdots}$ estimating the overall mean parameter). The SSE is most easily obtained by subtraction:

$$SSE = SST - SSBr - SSBc - SSTr$$

and has degrees of freedom (obtained by subtraction) $(r-1)(r-2)$.

The most important test is the test for significant treatments: $H_0 : \tau_k = 0, \text{for all } k$ versus $H_a :$ at least one treatment has an effect on the mean response.

8.3.1 Example: MPG experiment

The following MPG measurements were obtained using a Latin square design.

		Driver			
		1	2	3	4
Car	1	A = 24	B = 26	C = 16	D = 20
	2	B = 15	C = 26	D = 20	A = 16
	3	C = 17	D = 13	A = 20	B = 27
	4	D = 23	A = 15	B = 20	C = 25

```
mpg <- data.frame(car = c(1,2,3,4,1,2,3,4,1,2,3,4,1,2,3,4), driver = c(1,1,1,1,2,2,2,2,3,3,3,3,4,
mpg$car <- factor(mpg$car, levels = unique(mpg$car))
mpg$driver <- factor(mpg$driver, levels = unique(mpg$driver))
mpg$additive <- factor(mpg$additive, levels = unique(mpg$additive))

my.aov <- aov(mpg~additive + car+driver, data = mpg)
summary(my.aov)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## additive     3  29.69    9.90   0.241  0.865
## car          3  15.19    5.06   0.124  0.943
## driver       3  19.69    6.56   0.160  0.919
## Residuals    6 245.87   40.98
```

8.4 Relative efficiency of blocking

The efficiency of a Latin square design may be assessed relative to a CRD or an RCBD using only the row or only the column blocking variable. For comparison to and RCBD using the row block, we have the formula:

$$RE = \frac{MSE_{row}}{MSE_{LS}} \times \text{df correction}$$

where $MSE_{row} = \frac{MS_{row}+(r-1)MSE_{LS}}{r}$, where MSE_{LS} denotes the observed MSE from the Latin square experiment, MS_{row} denotes the oserved mean sum of squares of the row blocking variable from the Latin square experiment, and where

$$\text{df correction} = \frac{df_{E,LS}+1}{df_{E,RCBD}+1} \times \frac{df_{E,RCBD}+3}{df_{E,LS}+3} = \frac{r^2-3r+3}{r^2-2r+2} \times \frac{r^2-2r+4}{r^2-3r+5}.$$

8.4.1 Example: RE ignoring driver blocks

Neither blocking variable was particularly helpful. The RE relative to RCBD without driver is only about 74%.

```
MSE_row <- (6.56 + 3*40.98)/4
correction <- function(r)  (r^2 - 3*r+3)*(r^2 - 2*r+4)/((r^2-3*r+5)*(r^2 - 2*r+2))
RE <- (MSE_row / 40.98)*correction(4)
RE
```

```
## [1] 0.7373516
```

Chapter 9

Two-Way ANOVA

Two-way ANOVA models represent experiments with two categorical treatment variables and aim to quantify uncertainty about differences in mean response over the levels of these treatments. Superficially, these models appear similar to RCBDs, but neither categorical variable is a block, rather, the effects of both on the response are of interest. We could simply cross the levels of the two treatments and perform a one-way ANOVA, but that will make it challenging to sort out which treatment variable drives the differences in mean response. If we were to apply Tukey tests afterwards, we would have difficulties interpreting comparisons of different levels of different treatment variables. The main advantage of the two-way setup is the presence of explicit "interaction" effects for the crossed treatments. If these are insignificant in their effects on the mean response, then the model is simply additive, i.e., the effects of the two treatments are separable and therefore easily interpretable.

9.1 The Model

Let $i = 1, \ldots, a$ denote the number of levels of treatment variable A, $j = 1, \ldots, b$ denote the same for treatment variable B, and let $k = 1, \ldots, n$ denote the number of replications at each pair of crossed levels. We consider only balanced designs (for now). Then, the effects model may be written

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$$

where α_i and β_j are the "main effects" parameters for the treatment variables A and B and the $(\alpha\beta)_{ij}$ parameters are "interaction" effects representing synergistic effects between the two treatment variables—in other words, allowing for the treatment effects to be non-additive. As written, the model contains too many parameters—there are 1 + a + b + ab for a total of ab crossed treatment

mean responses. So, we need 1+a+b constraints. One choice is the baseline constraints in which we set $\alpha_a = \beta_b = (\alpha\beta)_{i,b} = (\alpha\beta)_{a,j} = 0$. This zeroes out $1 + 1 + a + b - 1$ parameters. (Note, there is a redundancy because $((\alpha\beta)_{a,b})$ appears twice in the constraint set, hence the minus 1.) It is also possible to specify sum-to-zero constraints.

9.1.1 Example: protein chemistry

An experiment is done in which minnows are fed diets with different amounts of zinc and copper supplements. The experimenters record the protein content of the minnows once harvested. Let the levels of copper be 0 and 100 and for zinc be 0, 100 and 200. Then, there are 6 treatment mean responses. Using the baseline constraint, we have the parameters $\mu, \alpha_1, \beta_1, \beta_2, (\alpha\beta)_{11}, (\alpha\beta)_{12}$.

9.2 Tests for interaction and main effects

The most important questions to be answered are: "do the treatments effect the mean response?" and "are those effects separable/additive?" We may answer these questions using tests of interaction and main effects. The test for interaction has null hypothesis $H_0 : (\alpha\beta)_{ij} = 0$, $\forall i, j$, versus the alternative that at least one interaction term is non-zero. If we fail to reject the null hypothesis, it means the effects of the treatments (if there are any) are separable/additive, i.e., the model is $Y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}$. This means we can interpret the treatment effects separately.
If we fail to reject the null hypothesis of the interaction test then we may move on to testing main effects: $H_0 : \alpha_i = 0$, $\forall i$ and $H_0 : \beta_j = 0$, $\forall j$ versus the alternatives that at least one main effect is non-zero. There is no point in doing these tests if there are interaction effects—in that case the treatments must have an effect on the mean response.
The sums of squares for testing interaction and main effects are given by the following:

$$SS_A = \sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n}(\bar{Y}_{i\cdot\cdot} - \bar{Y}_{\cdots})^2$$

$$SS_B = \sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n}(\bar{Y}_{\cdot j\cdot} - \bar{Y}_{\cdots})^2$$

$$SS_{Int} = \sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n}(\bar{Y}_{ij\cdot} - \bar{Y}_{i\cdot\cdot} - \bar{Y}_{\cdot j\cdot} + \bar{Y}_{\cdots})^2$$

$$SST = \sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{n}(Y_{ijk} - \bar{Y}_{\cdots})^2 \quad SSE = SST - SS_A - SS_B - SS_{Int}$$

The degrees of freedom associated with SS_A is $a-1$, for SS_B it is $b-1$, for SS_{Int} it is $(a-1)(b-1)$ and for SSE we find it by subtraction $(abn-1)-(a-1)-(b-1)-(a-1)(b-1)$. Then, the F test for interaction has test statistic

$$F_{Int} = \frac{SS_{Int}/(a-1)(b-1)}{SSE/df_{SSE}} \overset{H_0}{\sim} F_{(a-1)(b-1),df_{SSE}}.$$

Similarly, for testing the main effect A

$$F_A = \frac{SS_A/(a-1)}{SSE/df_{SSE}} \overset{H_0}{\sim} F_{a-1,df_{SSE}}.$$

9.2.1 Example: Protein and dietary metals

Experimenters raised minnows in 36 water tanks under different dietary regimes. The experimenters varied the levels of zinc and copper in the minnow's diets and recorded the average protein content of the minnows at harvest.

We can code the design matrix for a two way model under sum-two-zero constraints by hand and compute the estimated effects:

```
protein.df<- read.table('protein.txt')
colnames(protein.df) = c('Copper', 'Zinc', 'Protein')

Y <- c(201, 186, 173, 162, 115, 124, 163, 182, 184, 157, 114, 108)
X <- cbind(c(1,1,1,1,1,1,1,1,1,1,1,1),
           c(1,1,1,1,1,1,-1,-1,-1,-1,-1,-1),
           c(1,1,0,0,-1,-1,1,1,0,0,-1,-1),
           c(0,0,1,1,-1,-1,0,0,1,1,-1,-1),
           c(1,1,0,0,-1,-1,-1,-1,0,0,1,1),
           c(0,0,1,1,-1,-1,0,0,-1,-1,1,1))
X
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6]
##  [1,]    1    1    1    0    1    0
##  [2,]    1    1    1    0    1    0
##  [3,]    1    1    0    1    0    1
##  [4,]    1    1    0    1    0    1
##  [5,]    1    1   -1   -1   -1   -1
##  [6,]    1    1   -1   -1   -1   -1
##  [7,]    1   -1    1    0   -1    0
##  [8,]    1   -1    1    0   -1    0
##  [9,]    1   -1    0    1    0   -1
## [10,]    1   -1    0    1    0   -1
## [11,]    1   -1   -1   -1    1    1
## [12,]    1   -1   -1   -1    1    1
```

```
beta.hat <- solve(t(X)%*%X)%*%t(X)%*%Y
beta.hat
```

```
##             [,1]
## [1,] 155.750000
## [2,]   4.416667
## [3,]  27.250000
## [4,]  13.250000
## [5,]   6.083333
## [6,]  -5.916667
```

```
 solve(t(X)%*%X)%*%t(X)
```

```
##              [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
## [1,]  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333
## [2,]  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333
## [3,]  0.16666667  0.16666667 -0.08333333 -0.08333333 -0.08333333 -0.08333333
## [4,] -0.08333333 -0.08333333  0.16666667  0.16666667 -0.08333333 -0.08333333
## [5,]  0.16666667  0.16666667 -0.08333333 -0.08333333 -0.08333333 -0.08333333
## [6,] -0.08333333 -0.08333333  0.16666667  0.16666667 -0.08333333 -0.08333333
##              [,7]        [,8]        [,9]       [,10]       [,11]       [,12]
## [1,]  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333
## [2,] -0.08333333 -0.08333333 -0.08333333 -0.08333333 -0.08333333 -0.08333333
## [3,]  0.16666667  0.16666667 -0.08333333 -0.08333333 -0.08333333 -0.08333333
## [4,] -0.08333333 -0.08333333  0.16666667  0.16666667 -0.08333333 -0.08333333
## [5,] -0.16666667 -0.16666667  0.08333333  0.08333333  0.08333333  0.08333333
## [6,]  0.08333333  0.08333333 -0.16666667 -0.16666667  0.08333333  0.08333333
```

Note the the estimates are response contrasts. for example,

$\hat{\alpha}_1 = (1/6, 1/6, 1/6, 1/6, 1/6, 1/6, -1/6, -1/6, -1/6, -1/6, -1/6, -1/6)^\top Y = \bar{Y}_{1\cdot} - \bar{Y}_{2\cdot}$

Under the sum-to-zero constraints the effects estimates are the following linear combinations of marginal response means:

```
mean(Y)
```

```
## [1] 155.75
```

```
mean(Y[1:6]) - mean(Y)
```

```
## [1] 4.416667
```

```
mean(Y[c(1,2,7,8)]) - mean(Y)
```

```
## [1] 27.25
```

```
mean(Y[c(3,4,9,10)]) - mean(Y)
```

```
## [1] 13.25
```

```
mean(Y[c(1,2)]) - mean(Y[c(1,2,7,8)]) - mean(Y[1:6]) + mean(Y)
```

```
## [1] 6.083333
```

```
mean(Y[c(3,4)]) - mean(Y[c(3,4,9,10)]) - mean(Y[1:6]) + mean(Y)
```

```
## [1] -5.916667
```

Alternatively, we can compute the effects estimates in R using the options/contrasts statement to use sum-to-zero constraints:

```
protein = protein.df
colnames(protein) <- c("Copper", "Zinc", "Protein")
protein$Copper <- as.factor(protein$Copper)
protein$Zinc <- as.factor(protein$Zinc)
options(contrasts = c("contr.sum", "contr.sum"))
my.aov <- aov(Protein~Copper + Zinc + Copper*Zinc, data = protein)
summary(my.aov)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Copper       1    234     234   1.809 0.227264
## Zinc         2  10234    5117  39.537 0.000351 ***
## Copper:Zinc  2    288     144   1.113 0.387957
## Residuals    6    777     129
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
my.aov$coefficients
```

```
##   (Intercept)       Copper1         Zinc1         Zinc2 Copper1:Zinc1
##    155.750000      4.416667     27.250000     13.250000      6.083333
## Copper1:Zinc2
##     -5.916667
```

We can verify the t-tests in the lm output using our formulas for testing contrasts:

```
 c1<-c(1/6, 1/6, -1/12, -1/12, -1/12, -1/12, 1/6,1/6,-1/12, -1/12, -1/12, -1/12)
sum(c1^2)
```

```
## [1] 0.1666667
```

```
129/6
```

```
## [1] 21.5
```

```
sum(c1*protein$Protein)
```

```
## [1] 27.25
```

```
27.25/sqrt(21.5)
```

```
## [1] 5.876886
```

```
2*(1-pt(5.876886, 6))
```

```
## [1] 0.001075079
```

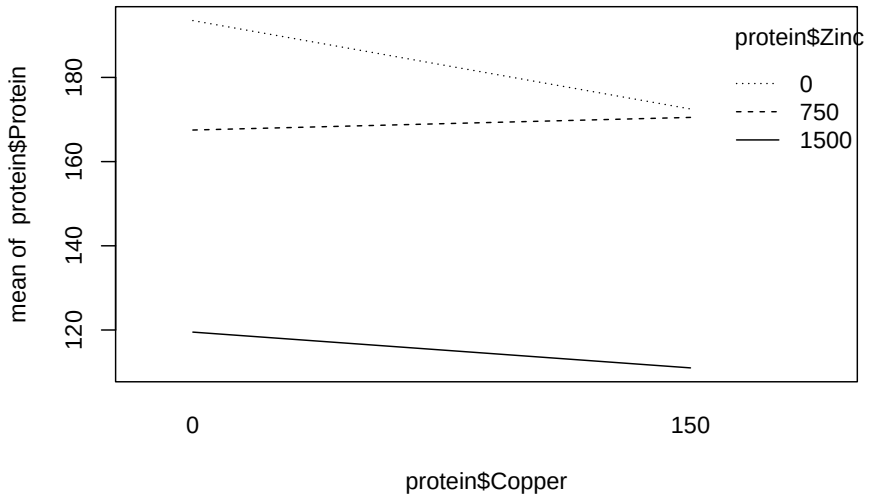
```
summary(lm(Protein~Copper + Zinc + Copper*Zinc, data = protein))
```

```
##
## Call:
## lm(formula = Protein ~ Copper + Zinc + Copper * Zinc, data = protein)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -13.5   -6.0    0.0    6.0   13.5
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    155.750      3.284  47.427 5.89e-09 ***
## Copper1          4.417      3.284   1.345  0.22726
## Zinc1           27.250      4.644   5.867  0.00108 **
## Zinc2           13.250      4.644   2.853  0.02907 *
## Copper1:Zinc1    6.083      4.644   1.310  0.23816
```

```
## Copper1:Zinc2   -5.917      4.644  -1.274  0.24980
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.38 on 6 degrees of freedom
## Multiple R-squared:  0.9327, Adjusted R-squared:  0.8766
## F-statistic: 16.62 on 5 and 6 DF,  p-value: 0.001854
```

R includes the "interaction.plot" function to graphically investigate interactions. The fact the lines only slightly cross in one plot supports no interaction. We also see the F test for interaction is not significant.

```
interaction.plot(protein$Copper, protein$Zinc, protein$Protein)
```




```
interaction.plot(protein$Zinc, protein$Copper, protein$Protein)
```


protein$Copper
0
150
mean of protein$Protein
120
140
160
180
0
750
1500
protein$Zinc

Chapter 10

Unbalanced data in Two-Way ANOVA

What happens if n_{ij} is not the same over crossed levels of the treatment factors? And, worse, what if some combination has zero replicates?!?

This is very common for two reasons: in observational studies researchers have less control over the number of subjects with certain levels of factors. Even when these are controlled through sampling or fixed in experimental studies there will be individuals that are lost/missing for some reason, and this ruins balance.

What goes wrong and what still works? (some of these things we haven't yet defined) - Comparisons using contrasts - still work! - Partial F tests in linear model - still work! - least-squares means - still work! - Sums of squares (you guessed it) - fails miserably!

Throughout we'll assume missing observations/lack of balance occurs at random, that is, does not have anything to do with treatments. If, however, a treatment level causes loss of observations, our analysis will have problems. You can imagine a treatment that causes side-effects in humans may cause "loss to follow-up" when taking before-after measures, because people will dislike the treatment and drop out of the study. So, independence of unbalance and treatment is not always a given.

10.1 Raw and Least-Squares Means

Consider a two-way ANOVA where factors A and B each have 2 levels, simplest possible setup. Given observations:

		Cell Means (Sample Sizes)	
		B	
		1	2
A	1	4.90(10)	2.79(4)
	2	8.14(2)	6.00(4)

When the sample sizes were all equal (balanced) sample means were apples-to-apples. Now, not so much. We can define the "raw sample means" (sample-size weighted averages) by $\overline{Y}_{1\cdot\cdot} = 4.30 = \frac{1}{14}(10 \times 4.9 + 4 \times 2.79)$. The raw means are misleading if the other factor has an effect, because B at level 1 is more represented in the raw mean average than B at level 2. The difference of raw means $\overline{Y}_{1\cdot\cdot} - \overline{Y}_{2\cdot\cdot}$ also contains an effect from B, which is hidden/disguised when we sum over levels of B. The raw means for factor A are defined $\mu^R_{i\cdot} = [\sum_j n_{ij}]^{-1} \sum_j n_{ij}\mu_{ij}$.

The "least-squares means" on the other hand, average the cell means without regard for sample size: mean of level 1 of A is $(2.79 + 4.9)/2 = 3.85$. The LS means for factor A are defined $\mu_{i\cdot} = \sum_j \mu_{ij}/b$ where b is the number of levels of B.

When the experiment is balanced, these are equal.

The estimators of the raw means for factor A are simply the raw sample means $\widehat{\mu}^R_{i\cdot} = [\sum_j n_{ij}]^{-1} \sum_j n_{ij}\overline{Y}_{ij}$. These have standard error

$$\text{s.e.}(\widehat{\mu}^R_{i\cdot}) = \sigma \left[\sum_j n_{ij}\right]^{-1/2}.$$

The estimated LS means have standard error

$$\text{s.e.}(\widehat{\mu}_{i\cdot}) = \sigma \sqrt{b^{-2} \sum_j n_{ij}^{-1}}.$$

"Simple effects" compare two levels of one factor at a fixed level of another factor. Differences of LS means are unweighted averages of simple effects:

$$\sum_j (\mu_{ij} - \mu_{i'j})/b$$

10.1.1 Example: Turbine data

The following data describes characteristics of turbine jet engines. Turbine engines feature many spinning blades. Both the position and rate of revolutions of the moving blades affect the behavior of the outflow of air generated by the

engines. The Strouhal number is one quantity that characterizes the oscillations of this flow of air. In the following data the Strouhal number is the response variable, and both rate and position of turbine blades are treatment factors.

The turbine data is balanced, but we'll remove a few rows to create imbalance, just for illustration.

```
strouhal.df <- read.csv('flow.csv')
strouhal.df
```

```
##    strouhal position rate
## 1  26016.65       -1    0
## 2  28254.84       -1    0
## 3  27413.45       -1    0
## 4  28556.40       -1    0
## 5  26029.52        1    0
## 6  28270.20        1    0
## 7  27370.98        1    0
## 8  28542.62        1    0
## 9  30359.70       -1    1
## 10 33191.74       -1    1
## 11 31487.19       -1    1
## 12 33111.07       -1    1
## 13 30066.25        1    1
## 14 32868.95        1    1
## 15 31757.56        1    1
## 16 33437.95        1    1
## 17 32189.43       -1    2
## 18 33782.31       -1    2
## 19 34090.11       -1    2
## 20 31098.27       -1    2
## 21 30994.56        1    2
## 22 31013.19        1    2
## 23 34892.13        1    2
## 24 33116.98        1    2
```

```
strouhal.df<-strouhal.df[-c(1,8,24),]
strouhal.df
```

```
##    strouhal position rate
## 2  28254.84       -1    0
## 3  27413.45       -1    0
## 4  28556.40       -1    0
## 5  26029.52        1    0
## 6  28270.20        1    0
```

```
## 7  27370.98        1    0
## 9  30359.70       -1    1
## 10 33191.74       -1    1
## 11 31487.19       -1    1
## 12 33111.07       -1    1
## 13 30066.25        1    1
## 14 32868.95        1    1
## 15 31757.56        1    1
## 16 33437.95        1    1
## 17 32189.43       -1    2
## 18 33782.31       -1    2
## 19 34090.11       -1    2
## 20 31098.27       -1    2
## 21 30994.56        1    2
## 22 31013.19        1    2
## 23 34892.13        1    2
```

```
strouhal.df$position<-as.factor(strouhal.df$position)
strouhal.df$rate<-as.factor(strouhal.df$rate)
```

Below are the crossed treatment mean responses, the LS means, and the raw means:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
by_trt <- group_by(strouhal.df, position, rate)
mean_resp <- summarise(by_trt, trt_mean = mean(strouhal))
```

```
## `summarise()` has grouped output by 'position'. You can override using the
## `.groups` argument.
```

```
mean_resp
```

```
## # A tibble: 6 x 3
```

```
## # Groups:   position [2]
##   position rate  trt_mean
##   <fct>    <fct>    <dbl>
## 1 -1       0       28075.
## 2 -1       1       32037.
## 3 -1       2       32790.
## 4 1        0       27224.
## 5 1        1       32033.
## 6 1        2       32300.
```

```
mean_resp<-as.matrix(mean_resp[,3])
LS_mean_pos <- c(mean(mean_resp[1:3,1]), mean(mean_resp[4:6,1]))
LS_mean_pos
```

```
## [1] 30967.45 30518.73
```

```
LS_mean_rate <- c(mean(mean_resp[c(1,4),1]), mean(mean_resp[c(2,5),1]), mean(mean_resp[c(3,6),1]))
LS_mean_rate
```

```
## [1] 27649.23 32035.05 32544.99
```

```
by_pos <- group_by(strouhal.df, position)
raw_mean_pos <- summarise(by_pos, trt_mean = mean(strouhal))
raw_mean_pos
```

```
## # A tibble: 2 x 2
##   position trt_mean
##   <fct>       <dbl>
## 1 -1         31230.
## 2 1          30670.
```

```
by_rate <- group_by(strouhal.df, rate)
raw_mean_rate <- summarise(by_rate, trt_mean = mean(strouhal))
raw_mean_rate
```

```
## # A tibble: 3 x 2
##   rate  trt_mean
##   <fct>    <dbl>
## 1 0       27649.
## 2 1       32035.
## 3 2       32580
```

It is also convenient to compute the LS means using built-in functions from the package emmeans:. The "emmean" stands for estimated marginal mean. These are calculated based on a model. If we use the two way model with interaction, then the emmeans are the LS means.

```
library(emmeans)
emmeans(lm(strouhal~position+rate+position*rate, data = strouhal.df), 'position')
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
##  position emmean  SE df lower.CL upper.CL
##  -1        30967 441 15    30028    31906
##  1         30519 462 15    29534    31504
##
## Results are averaged over the levels of: rate
## Confidence level used: 0.95
```

```
emmeans(lm(strouhal~position+rate+position*rate, data = strouhal.df), 'rate')
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
##  rate emmean  SE df lower.CL upper.CL
##  0     27649 591 15    26389    28909
##  1     32035 512 15    30944    33126
##  2     32545 553 15    31367    33723
##
## Results are averaged over the levels of: position
## Confidence level used: 0.95
```

Next, lets compute CIs for contrasts of LS means. For example, consider comparing mean Strouhal values at rate level 0 versus rate level 1. The point estimate is

$$\overline{Y}_{\cdot,0} - \overline{Y}_{\cdot,1}$$

the sample LS mean at level zero of rate summed over position minus the sample LS mean at level one of rate summed over position. This difference of sample means has the following standard error:

$$V(\overline{Y}_{\cdot,0} - \overline{Y}_{\cdot,1}) = V(\tfrac{1}{2}(\overline{Y}_{-1,0} + \overline{Y}_{1,0}) - \tfrac{1}{2}(\overline{Y}_{-1,1} + \overline{Y}_{1,1}))// \quad = \sigma^2 \tfrac{1}{4}(1/3 + 1/3 + 1/4 + 1/4)$$

using the cell sample sizes calculated in R below.

```
by_trt <- group_by(strouhal.df, position, rate)
n_resp <- summarise(by_trt, trt_mean = n())
```

```
## `summarise()` has grouped output by 'position'. You can override using the
## `.groups` argument.
```

```
n_resp
```

```
## # A tibble: 6 x 3
## # Groups:   position [2]
##   position rate  trt_mean
##   <fct>    <fct>    <int>
## 1 -1       0            3
## 2 -1       1            4
## 3 -1       2            4
## 4 1        0            3
## 5 1        1            4
## 6 1        2            3
```

Then, a CI for the difference in LS means is given by

$$\left(\overline{Y}_{\cdot,0} - \overline{Y}_{\cdot,1} \pm t_{1-\alpha/2,n-p}\sqrt{MSE\tfrac{1}{4}(1/3+1/3+1/4+1/4)}\right)$$

```
my.lm <- summary(lm(strouhal~position+rate+position*rate, data = strouhal.df))

c(LS_mean_rate[1] - LS_mean_rate[2] - qt(0.975, my.lm$df[2])*my.lm$sigma*sqrt((1/4)* (1/3 + 1/3 +
```

```
## [1] -6052.374 -2719.265
```

10.2 Partial F tests and Type III SS

To test for interaction and main effects in unbalanced, to-way ANOVA, we cannot use the same sums of squares formulas as before. These do not work for unbalanced data. Essentially, they result in comparisons of raw, rather than LS means.

Instead, we test for interaction and main effects using *partial F tests*. Think of the Gauss-Markov notation $Y = X\beta + \epsilon$. A partial F test is used to test the hypothesis H_0 : a certain set of β_j coefficients is zero. The set could be as small as one coefficient, or as large as all the coefficients except for the intercept. Under the null hypothesis, inclusion in the model of the covariates corresponding to these coefficients does not substantially change the predicted

responses $\hat{Y}$; and, as a result, does not substantially reduce the sum of squared residuals. Therefore, the intuitive test of this null hypothesis is a comparison of SSE under models including versus excluding the given set of covariates:

$$F = \frac{[SSE(\text{reduced model}) - SSE(\text{full model})]/\text{number of excluded coefficients}}{MSE(\text{full model})}$$

Under H_0 the test statistic F has an F distribution with numerator degrees of freedom equal to the number of excluded coefficients and denominator degrees of freedom equal to $n - p$ where the full model coefficient vector β has dimension $p \times 1$. SAS, and certain R functions, will report *Type III sums of squares* for unbalanced experiments. These are simply defined by partial F tests. For example, the Type III SS for factor A in a two-way ANOVA with interaction is simply the difference in the sum of squared residuals for the full linear model corresponding to the two-way ANOVA versus the reduced linear model where the columns of X corresponding to the factor A main effects have been removed. Therefore, tests based on Type III sums of squares are exactly partial F tests.

10.2.1 Example: Turbine Data tests

We can perform partial F tests by fitting reduced and full linear models and comparing sums of squared residuals. The R package *car* will perform these tests automatically using the function *Anova*. The following code computes a tests for the main effect of rate, both "by hand" by fitting full and reduced models, and using the Anova function. Note the equivalence of F statistics.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
Anova(lm(strouhal~position + rate+position*rate, data = strouhal.df), type = 'III')
```

```
## Anova Table (Type III tests)
##
## Response: strouhal
##                   Sum Sq Df   F value    Pr(>F)
## (Intercept)   2364599469  1 1128.1201 1.564e-15 ***
## position         1087144  1    0.5187  0.482485
## rate            42206544  2   10.0681  0.001689 **
## position:rate     631584  2    0.1507  0.861425
## Residuals       31440793 15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm(strouhal~position + rate+position*rate, data = strouhal.df))
```

```
##
## Call:
## lm(formula = strouhal ~ position + rate + position * rate, data = strouhal.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1966.4 -1194.0   147.4  1046.6  2592.2
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      28074.9      835.9  33.587 1.56e-15 ***
## position1         -851.3     1182.1  -0.720 0.482485
## rate1             3962.5     1105.8   3.584 0.002716 **
## rate2             4715.1     1105.8   4.264 0.000679 ***
## position1:rate1    846.6     1563.8   0.541 0.596202
## position1:rate2    361.3     1618.7   0.223 0.826403
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1448 on 15 degrees of freedom
## Multiple R-squared:  0.7511, Adjusted R-squared:  0.6681
## F-statistic: 9.053 on 5 and 15 DF,  p-value: 0.000396
```

```
full.model <- lm(strouhal~position + rate+position*rate, data = strouhal.df)
SSE.full <- sum(full.model$residuals^2)
MSE <- SSE.full/(nrow(strouhal.df)-6)

X.R <- model.matrix(strouhal~position + rate+position*rate, data = strouhal.df)
```

```
X.R <- X.R[,-c(3,4)]

beta.hat.R <- solve(t(X.R)%*%X.R)%*%t(X.R)%*%strouhal.df$strouhal
pred.R <- X.R%*%beta.hat.R
SSE.R <- sum((strouhal.df$strouhal - pred.R)^2)

F <- ((SSE.R - SSE.full)/(2)) / (SSE.full / (nrow(X.R)-6))
F
```

```
## [1] 10.0681
```

10.3 Follow-up tests

From the Anova output in the above example we see that the main effect for rate is significant. Since rate has 3 factors, it makes sense to perform Tukey-corrected pairwise comparisons to better understand which levels of rate drive the significance of the effect.

For balanced experiments we typically use the R function TukeyHSD, which takes an aov object as its argument. But, since aov objects are based on Type I SS (raw means) these pairwise comparisons are not the comparisons we want to make. Rather, we want to make pairwise comparisons of LS means.

We can compute these by hand , as we did above for uncorrected pairwise comparisons. The only difference is that we replace the Student's t quantile in the CI calculation by the upper $1-\alpha$ quantile of Tukey's Studentized range distribution, multiplied by $1/\sqrt{2}$.

10.3.1 Example: Turbine data Tukey-corrected pairwise comparisons

```
n_resp
```

```
## # A tibble: 6 x 3
## # Groups:   position [2]
##   position rate  trt_mean
##   <fct>    <fct>    <int>
## 1 -1       0            3
## 2 -1       1            4
## 3 -1       2            4
## 4 1        0            3
## 5 1        1            4
## 6 1        2            3
```

```
c(LS_mean_rate[1] - LS_mean_rate[2] - sqrt(1/2)*qtukey(0.95,3, my.lm$df[2])*my.lm$sigma*sqrt((1/4
```

```
## [1] -6416.750 -2354.889
```

```
1-ptukey(abs(LS_mean_rate[1] - LS_mean_rate[2])/(sqrt(1/2)*my.lm$sigma*sqrt((1/4)* (1/3 + 1/3 + 1
```

```
## [1] 0.0001390661
```

```
c(LS_mean_rate[1] - LS_mean_rate[3] - sqrt(1/2)*qtukey(0.95,3, my.lm$df[2])*my.lm$sigma*sqrt((1/4
```

```
## [1] -6997.976 -2283.607
```

```
1-ptukey(abs(LS_mean_rate[1] - LS_mean_rate[3])/(sqrt(1/2)*my.lm$sigma*sqrt((1/4)* (1/3 + 1/3 + 1
```

```
## [1] 6.250909e-05
```

```
c(LS_mean_rate[2] - LS_mean_rate[3] - sqrt(1/2)*qtukey(0.95,3, my.lm$df[2])*my.lm$sigma*sqrt((1/4
```

```
## [1] -2466.998 -2428.766
```

```
1-ptukey(abs(LS_mean_rate[2] - LS_mean_rate[3])/(sqrt(1/2)*my.lm$sigma*sqrt((1/4)* (1/4 + 1/4 + 1
```

```
## [1] 0.7802542
```

```
library(emmeans)
em.strouhal<-emmeans(lm(strouhal~position + rate+position*rate, data = strouhal.df), "rate")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
pairs(em.strouhal, adjust = 'tukey')
```

```
##  contrast      estimate  SE df t.ratio p.value
##  rate0 - rate1    -4386 782 15  -5.609  0.0001
##  rate0 - rate2    -4896 809 15  -6.049  0.0001
##  rate1 - rate2     -510 753 15  -0.677  0.7803
##
## Results are averaged over the levels of: position
## P value adjustment: tukey method for comparing a family of 3 estimates
```

10.4 Unbalanced data and Simpson's Paradox

The following toy example of unbalanced data helps illustrate the difference between using raw and LS means. The LS mean response for A level 1 is larger (-0.3) than the LS mean response for A level 2 (-0.7). And, the mean response when A is level 1 is larger than when A is level 2 for every level of B. However, for raw means, the reverse is true (-1.975 vs. 0.225)! If we look at the data we see this is due to the fact more samples are observed for the levels of B when A is level 1 and the response is lowest (10) compared to when it is highest (5). Basically, the sampling is biased towards levels of B where the A level 1 response is low and the A level 2 response is high. The raw means are not capturing a real difference in effect of A; rather, they are capturing a difference in sample weighting.

This is an example of **Simpson's Paradox** where the direction of an effect may change if effects are weighted by sample size.

```
Y1 <- (1:5)+0.1
Y2 <- (1:5)
Y3 <- c((1:5)-10,(1:5)-10)
Y4 <- c((1:5),1:5)
Y5 <- (1:5)-0.1
Y6 <- (1:5)-11

A <- c(rep(1,20),rep(2,20))
B <- c(rep(1,5),rep(2,5), rep(3,10),rep(1,10),rep(2,5), rep(3,5))
Y <- c(Y1,Y2,Y3,Y4,Y5,Y6)
data.df <- data.frame(Y=Y,A=as.factor(A),B=as.factor(B))

by_trt <- group_by(data.df, A,B)
summarise(by_trt, trt_mean = mean(Y))
```

```
## `summarise()` has grouped output by 'A'. You can override using the `.groups`
## argument.
```

```
## # A tibble: 6 x 3
## # Groups:   A [2]
##   A     B     trt_mean
##   <fct> <fct>    <dbl>
## 1 1     1          3.1
## 2 1     2          3
## 3 1     3         -7
## 4 2     1          3
## 5 2     2          2.9
## 6 2     3         -8
```

```
emmeans(lm(Y~A+B+A*B, data = data.df), "A")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
##  A emmean    SE df lower.CL upper.CL
##  1   -0.3 0.362 34    -1.03   0.4348
##  2   -0.7 0.362 34    -1.43   0.0348
##
## Results are averaged over the levels of: B
## Confidence level used: 0.95
```

```
emmeans(lm(Y~A+B+A*B, data = data.df), "B")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
##  B emmean    SE df lower.CL upper.CL
##  1   3.05 0.420 34     2.20     3.90
##  2   2.95 0.485 34     1.96     3.94
##  3  -7.50 0.420 34    -8.35    -6.65
##
## Results are averaged over the levels of: A
## Confidence level used: 0.95
```

```
mean(data.df$Y[data.df$A==1])
```

```
## [1] -1.975
```

```
mean(data.df$Y[data.df$A==2])
```

```
## [1] 0.225
```

```
mean(data.df$Y[data.df$B==1])
```

```
## [1] 3.033333
```

```
mean(data.df$Y[data.df$B==2])
```

```
## [1] 2.95
```

```
mean(data.df$Y[data.df$B==3])
```

```
## [1] -7.333333
```

Chapter 11

Missing Cells in Two-Way ANOVA

In this section we consider how to analyze a two-factor experiment when one crossed treatment is unobserved. If there is no interaction, then this is not problem. But, when interaction is present the lack of observations in a crossed treatment group makes it challenging to assess those interactions. There are, essentially, two ways to proceed: 1) use a one-way model, treating the observed crossed treatments as the levels; or, 2) do a partial two-way analysis.

Below, we illustrate these two approaches in the context of a horticulture experiment about jalapenos.

11.1 ANOVA with missing "at random" cell

Scoville (spiciness) measure of jalapeno peppers grown under 3 different watering regimens and 3 different levels of sun exposure.

Unbalanced treatment groups with a missing crossed treatment, sun level 3 with water level 3.

```
my.data
```

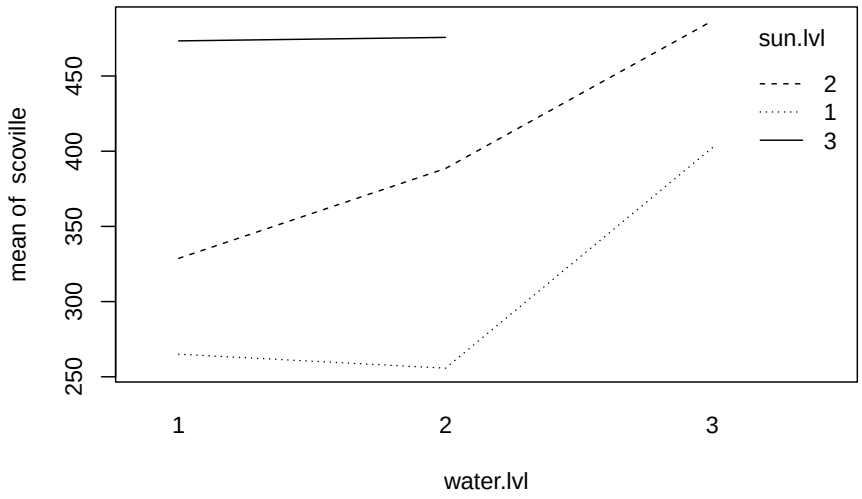
```
##    scoville water.lvl sun.lvl
## 1       259         1       1
## 2       271         1       1
## 3       289         2       1
## 4       255         2       1
## 5       361         2       1
```

```
## 6       118         2       1
## 7       463         3       1
## 8       372         3       1
## 9       372         3       1
## 10      208         1       2
## 11      288         1       2
## 12      482         1       2
## 13      337         1       2
## 14      452         2       2
## 15      325         2       2
## 16      582         3       2
## 17      411         3       2
## 18      467         3       2
## 19      512         1       3
## 20      430         1       3
## 21      478         1       3
## 22      646         2       3
## 23      436         2       3
## 24      345         2       3
```

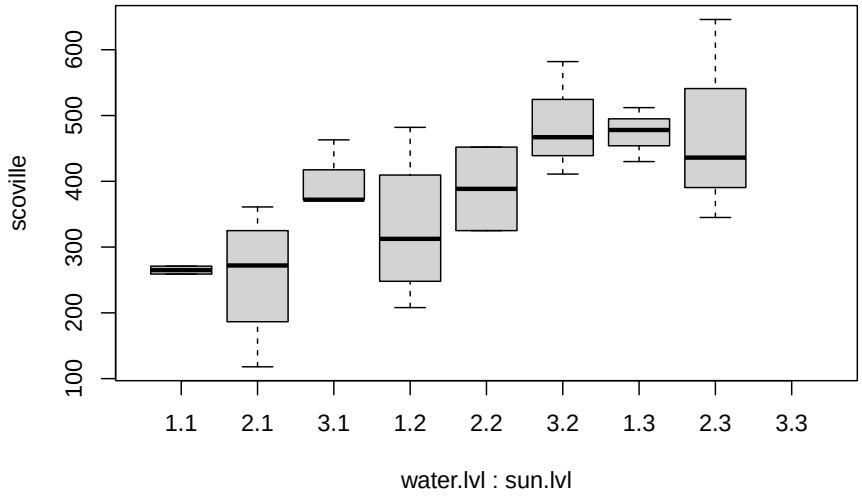
```
kable(aggregate(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data, FUN=mean)
```

water.lvl	sun.lvl	scoville
1	1	265.0000
2	1	255.7500
3	1	402.3333
1	2	328.7500
2	2	388.5000
3	2	486.6667
1	3	473.3333
2	3	475.6667

```
interaction.plot(water.lvl, sun.lvl,  scoville, data = my.data)
```




```
boxplot(scoville~water.lvl+ sun.lvl, data = my.data)
```

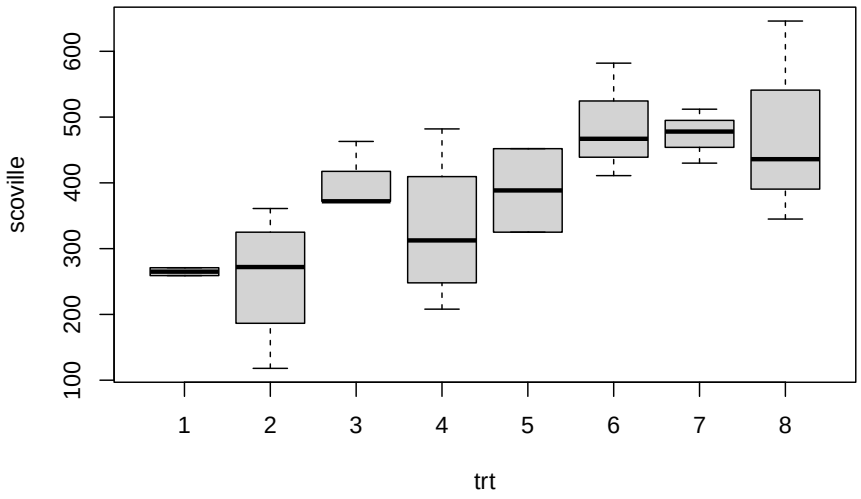


11.2 Do any of the treatments matter?

Can test this two equivalent ways: 1. fit a one way anova (ignoring that there are two treatment variables) and test if the treatments are different. 2. fit a linear model and perform the model F test comparing the full model to the intercept only model.

The linear model method is slightly more challenging to implement because of the additional constraint imposed by the missing cell. We have to modify the design matrix from the model.matrix function, or define it by hand.

One way anova below. Notice it says the treatments are not all the same; their mean responses significantly differ for at least one treatment compared to the other 7. One approach would be to follow this test up with tests of contrasts or pairwise comparisons, corrected using Scheff'e or Tukey. Since the data is unbalanced, we should do these by hand.




```
my.data.oneway
```

```
##    scoville trt
## 1       259   1
## 2       271   1
## 3       289   2
## 4       255   2
## 5       361   2
## 6       118   2
```

```
## 7       463   3
## 8       372   3
## 9       372   3
## 10      208   4
## 11      288   4
## 12      482   4
## 13      337   4
## 14      452   5
## 15      325   5
## 16      582   6
## 17      411   6
## 18      467   6
## 19      512   7
## 20      430   7
## 21      478   7
## 22      646   8
## 23      436   8
## 24      345   8
```

```
library(car)
Anova(lm(scoville~trt, data = my.data.oneway), type = 'III')
```

```
## Anova Table (Type III tests)
##
## Response: scoville
##              Sum Sq Df  F value   Pr(>F)
## (Intercept) 3339450  1 354.1618 2.44e-12 ***
## trt          188009  7   2.8484   0.0393 *
## Residuals    150867 16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

11.3 Fit a linear model with additional constraints

The other option for testing if any treatments matter is to fit a linear model. However, we cannot estimate all the parameters of the effects model due to missing the combination of sun level 3 and water level 3. We can only estimate 8 parameters: intercept, 2 for sun, 2 for water, and 3 (rather than 4) interactions. The way to deal with this is to find the full model matrix and then omit the last column corresponding to the interaction that is not estimable. IF we just use the lm() function, we'll get an error, but it will still give us some useful output. Again, we end up having to do some things by hand.

```
options(contrasts = c('contr.sum', 'contr.sum'))
my.lm<-lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data)
summary(my.lm)
```

```
##
## Call:
## lm(formula = scoville ~ water.lvl + sun.lvl + water.lvl * sun.lvl,
##     data = my.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -137.750  -41.396   -3.375   44.167  170.333
##
## Coefficients: (1 not defined because of singularities)
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)            386.11      41.79   9.240 8.15e-08 ***
## water.lvl1             -30.42      48.55  -0.626    0.540
## water.lvl2             -12.81      52.02  -0.246    0.809
## sun.lvl1               -78.42      48.55  -1.615    0.126
## sun.lvl2                15.19      52.02   0.292    0.774
## water.lvl1:sun.lvl1    -12.28      49.44  -0.248    0.807
## water.lvl2:sun.lvl1    -39.14      68.02  -0.575    0.573
## water.lvl1:sun.lvl2    -42.14      68.02  -0.619    0.544
## water.lvl2:sun.lvl2        NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 97.1 on 16 degrees of freedom
## Multiple R-squared:  0.5548, Adjusted R-squared:   0.36
## F-statistic: 2.848 on 7 and 16 DF,  p-value: 0.0393
```

```
my.lm$coefficients
```

```
##         (Intercept)          water.lvl1          water.lvl2            sun.lvl1
##           386.11111           -30.41667           -12.80556           -78.41667
##            sun.lvl2 water.lvl1:sun.lvl1 water.lvl2:sun.lvl1 water.lvl1:sun.lvl2
##            15.19444           -12.27778           -39.13889           -42.13889
## water.lvl2:sun.lvl2
##                  NA
```

```
X <- model.matrix(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data)
X
```

```
##    (Intercept) water.lvl1 water.lvl2 sun.lvl1 sun.lvl2 water.lvl1:sun.lvl1
```

```
## 1            1          1          0        1        0                  1
## 2            1          1          0        1        0                  1
## 3            1          0          1        1        0                  0
## 4            1          0          1        1        0                  0
## 5            1          0          1        1        0                  0
## 6            1          0          1        1        0                  0
## 7            1         -1         -1        1        0                 -1
## 8            1         -1         -1        1        0                 -1
## 9            1         -1         -1        1        0                 -1
## 10           1          1          0        0        1                  0
## 11           1          1          0        0        1                  0
## 12           1          1          0        0        1                  0
## 13           1          1          0        0        1                  0
## 14           1          0          1        0        1                  0
## 15           1          0          1        0        1                  0
## 16           1         -1         -1        0        1                  0
## 17           1         -1         -1        0        1                  0
## 18           1         -1         -1        0        1                  0
## 19           1          1          0       -1       -1                 -1
## 20           1          1          0       -1       -1                 -1
## 21           1          1          0       -1       -1                 -1
## 22           1          0          1       -1       -1                  0
## 23           1          0          1       -1       -1                  0
## 24           1          0          1       -1       -1                  0
##    water.lvl2:sun.lvl1 water.lvl1:sun.lvl2 water.lvl2:sun.lvl2
## 1                    0                   0                   0
## 2                    0                   0                   0
## 3                    1                   0                   0
## 4                    1                   0                   0
## 5                    1                   0                   0
## 6                    1                   0                   0
## 7                   -1                   0                   0
## 8                   -1                   0                   0
## 9                   -1                   0                   0
## 10                   0                   1                   0
## 11                   0                   1                   0
## 12                   0                   1                   0
## 13                   0                   1                   0
## 14                   0                   0                   1
## 15                   0                   0                   1
## 16                   0                  -1                  -1
## 17                   0                  -1                  -1
## 18                   0                  -1                  -1
## 19                   0                  -1                   0
## 20                   0                  -1                   0
## 21                   0                  -1                   0
```

```
## 22                  -1                   0                  -1
## 23                  -1                   0                  -1
## 24                  -1                   0                  -1
## attr(,"assign")
## [1] 0 1 1 2 2 3 3 3 3
## attr(,"contrasts")
## attr(,"contrasts")$water.lvl
## [1] "contr.sum"
##
## attr(,"contrasts")$sun.lvl
## [1] "contr.sum"
```

```
X <- X[,-9]
beta.hat <- solve(t(X)%*%X)%*%t(X)%*%my.data$scoville
beta.hat
```

```
##                          [,1]
## (Intercept)         386.11111
## water.lvl1          -30.41667
## water.lvl2          -12.80556
## sun.lvl1            -78.41667
## sun.lvl2             15.19444
## water.lvl1:sun.lvl1 -12.27778
## water.lvl2:sun.lvl1 -39.13889
## water.lvl1:sun.lvl2 -42.13889
```

```
Y.pred <-X%*%beta.hat
SSE <- sum((my.data$scoville-Y.pred)^2)
MSE <- SSE / (24 - 8)
MSE
```

```
## [1] 9429.167
```

11.4 General Linear Test for interaction

The linear model we just developed, with the additional constraint, allows us to perform more tests. We can use this model to perform General linear F tests for interaction and main effects.

The first estimated interaction parameter

$$1/3 * [\mu_{11} - \mu_{13} - (\mu_{31} - \mu_{32}) - (\mu_{22} - \mu_{23})]$$

(265.0000 - 473.3333 - (402.3333 - 486.6667) - (388.5000 - 475.6667))/3 = -12.27778

The test for interaction does not reject the null hypothesis of no interaction. Note, we cannot feed the Anova function the fitted linear model; it will produce an error because of the inability to fit all 9 parameters. So, we had to do this "by hand".

```
options(contrasts = c('contr.sum', 'contr.sum'))
aggregate(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data, FUN=mean)
```

```
##   water.lvl sun.lvl scoville
## 1         1       1 265.0000
## 2         2       1 255.7500
## 3         3       1 402.3333
## 4         1       2 328.7500
## 5         2       2 388.5000
## 6         3       2 486.6667
## 7         1       3 473.3333
## 8         2       3 475.6667
```

```
X <- model.matrix(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data)
X.R <- X[,1:5]  # no interaction parameters
beta.hat.R <- solve(t(X.R)%*%X.R)%*%t(X.R)%*%my.data$scoville
beta.hat.R
```

```
##                   [,1]
## (Intercept)  407.58333
## water.lvl1   -54.98160
## water.lvl2   -38.85173
## sun.lvl1    -102.70887
## sun.lvl2     -11.12446
```

```
Y.pred.R <-X.R%*%beta.hat.R
SSE.R <- sum((my.data$scoville-Y.pred.R)^2)
F <- ((SSE.R - SSE)/(3)) / MSE
1-pf(F,3,16)
```

```
## [1] 0.9381543
```

```
#library(car)
#Anova(lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data), type = 'III')
```

11.5 Partial analysis for interaction

An alternative approach is to break the ANOVA into "overlapping" pieces that have no missing cells.

Break down the table into two overlapping pieces, one with no sun level 3 and one with no water level 3. Let's work with the no sun level 3 table. The idea is the same for the other table. We can test for interaction and main effects within this table. Now, there are 6 treatments, so we estimate an intercept, two water parameters, 1 sun parameter, and 2 interaction parameters. The first interaction term (no sun level 3 table) is estimating

$$\frac{1}{3}(\mu_{11}-\mu_{12})-\frac{1}{6}(\mu_{21}-\mu_{22})-\frac{1}{6}(\mu_{31}-\mu_{32})$$

$265/3-255.7500/6-402.3333/6-328.7500/3+388.500/6+486.6667/6 = 14.93057$

Compare this interaction term to the linear model with the additional constraint. I think it seems like a more interpretable interaction term, which I believe is why some people like the partial analysis.

```
options(contrasts = c('contr.sum', 'contr.sum'))
set.seed(12345)
water.lvl <- as.factor(1+c(0,0,1,1,1,1,2,2,2,0,0,0,0,1,1,2,2,2))
sun.lvl <- as.factor(  1+c(0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1))
scoville <- round(rnorm(18, 100*(as.numeric(water.lvl)+as.numeric(sun.lvl)),100),0)
my.data1 <- data.frame(scoville, water.lvl, sun.lvl)

aggregate(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1, FUN=mean)
```

```
##   water.lvl sun.lvl scoville
## 1         1       1 265.0000
## 2         2       1 255.7500
## 3         3       1 402.3333
## 4         1       2 328.7500
## 5         2       2 388.5000
## 6         3       2 486.6667
```

```
library(car)
Anova(lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1),type = 'III')
```

```
## Anova Table (Type III tests)
##
## Response: scoville
##                    Sum Sq Df  F value    Pr(>F)
## (Intercept)       2088060  1 251.0408 2.075e-09 ***
## water.lvl           71684  2   4.3092   0.03887 *
## sun.lvl             36400  1   4.3763   0.05837 .
## water.lvl:sun.lvl    3360  2   0.2020   0.81984
## Residuals           99811 12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1))
```

```
##
## Call:
## lm(formula = scoville ~ water.lvl + sun.lvl + water.lvl * sun.lvl,
##     data = my.data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -137.750  -38.146   -3.375   53.812  153.250
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           354.50      22.37  15.844 2.07e-09 ***
## water.lvl1            -57.62      31.94  -1.804   0.0964 .
## water.lvl2            -32.38      31.94  -1.013   0.3308
## sun.lvl1              -46.81      22.37  -2.092   0.0584 .
## water.lvl1:sun.lvl1    14.93      31.94   0.467   0.6486
## water.lvl2:sun.lvl1   -19.57      31.94  -0.613   0.5516
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91.2 on 12 degrees of freedom
## Multiple R-squared:  0.5438, Adjusted R-squared:  0.3538
## F-statistic: 2.861 on 5 and 12 DF,  p-value: 0.06299
```

```
X <- model.matrix(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1)
X
```

```
##    (Intercept) water.lvl1 water.lvl2 sun.lvl1 water.lvl1:sun.lvl1
## 1            1          1          0        1                    1
## 2            1          1          0        1                    1
## 3            1          0          1        1                    0
## 4            1          0          1        1                    0
## 5            1          0          1        1                    0
## 6            1          0          1        1                    0
## 7            1         -1         -1        1                   -1
## 8            1         -1         -1        1                   -1
## 9            1         -1         -1        1                   -1
## 10           1          1          0       -1                   -1
## 11           1          1          0       -1                   -1
## 12           1          1          0       -1                   -1
## 13           1          1          0       -1                   -1
## 14           1          0          1       -1                    0
```

```
## 15           1          0          1       -1                    0
## 16           1         -1         -1       -1                    1
## 17           1         -1         -1       -1                    1
## 18           1         -1         -1       -1                    1
##    water.lvl2:sun.lvl1
## 1                    0
## 2                    0
## 3                    1
## 4                    1
## 5                    1
## 6                    1
## 7                   -1
## 8                   -1
## 9                   -1
## 10                   0
## 11                   0
## 12                   0
## 13                   0
## 14                  -1
## 15                  -1
## 16                   1
## 17                   1
## 18                   1
## attr(,"assign")
## [1] 0 1 1 2 3 3
## attr(,"contrasts")
## attr(,"contrasts")$water.lvl
## [1] "contr.sum"
##
## attr(,"contrasts")$sun.lvl
## [1] "contr.sum"
```

```
solve(t(X)%*%X)%*%t(X)%*%my.data1$scoville
```

```
##                           [,1]
## (Intercept)          354.50000
## water.lvl1           -57.62500
## water.lvl2           -32.37500
## sun.lvl1             -46.80556
## water.lvl1:sun.lvl1   14.93056
## water.lvl2:sun.lvl1  -19.56944
```

11.6 Tests for main effects in the partial table

(no sun level 3) We can match the main effects tests in the ANOVA (type 3) table by comparing LS means.

```
# contrast test for sun
aggregate(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1, FUN=mean)
```

```
##   water.lvl sun.lvl scoville
## 1         1       1 265.0000
## 2         2       1 255.7500
## 3         3       1 402.3333
## 4         1       2 328.7500
## 5         2       2 388.5000
## 6         3       2 486.6667
```

```
MSE <- sum(lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1)$residuals^2)/(18-6)
MSE
```

```
## [1] 8317.611
```

```
(((265+255.75+402.333)/3)-((328.75+388.5+486.667)/3))/sqrt(MSE * 2*(1/9 * (1/2+1/4+1/3)))
```

```
## [1] -2.091965
```

```
F<-((((265+255.75+402.333)/3)-((328.75+388.5+486.667)/3))/sqrt(MSE * 2*(1/9 * (1/2+1/4+1/3)))
)^2
F
```

```
## [1] 4.376316
```

```
1-pf(F,1,12)
```

```
## [1] 0.05837152
```

These tests for main effects should be constructed using the MSE from the full linear model fitted above rather than the partial tables. There is not much of a difference in the conclusion, but the partial table seems to underestimate the variance because it excludes sun level 3 which has larger responses.

```
# contrast test for sun
aggregate(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data1, FUN=mean)
```

```
##   water.lvl sun.lvl scoville
## 1         1       1 265.0000
## 2         2       1 255.7500
## 3         3       1 402.3333
## 4         1       2 328.7500
## 5         2       2 388.5000
## 6         3       2 486.6667
```

```
MSE <- sum(lm(scoville~water.lvl+sun.lvl+water.lvl*sun.lvl, data = my.data)$residuals^2
MSE
```

```
## [1] 9429.167
```

```
(((265+255.75+402.333)/3)-((328.75+388.5+486.667)/3))/sqrt(MSE * 2*(1/9 * (1/2+1/4+1/3)
```

```
## [1] -1.964794
```

```
F <- ((((265+255.75+402.333)/3)-((328.75+388.5+486.667)/3))/sqrt(MSE * 2*(1/9 * (1/2+1/
)^2
F
```

```
## [1] 3.860415
```

```
1-pf(F,1,16)
```

```
## [1] 0.06704815
```

Chapter 12

Strategy for Analysis in Two-Factor models

In this section we synthesize the main points from our discussion of two-way ANOVA to outline a general strategy for analyzing data from two-factor experiments.

For experiments in which it is known that factors do not interact, comparisons for treatments are made on the basis of factor-level means $\mu_{i\cdot}$ and $\mu_{\cdot j}$: 1. Fit the additive model

$$Y_{ijk} = \mu + \alpha_i + \tau_j + \epsilon_{jik},$$

and test for significance of the model using a partial F test comparing the intercept-only model to the full model. If the hypothesis H_0 : only the intercept is nonzero is rejected, then move to step 2. 2. Test for significance of Factor A main effects, $H_0 : \alpha_i = 0$, for all i versus H_a : not all $\alpha_i = 0$. 3. If we reject H_0, move to 4. Otherwise, perform follow-up comparisons of the factor level means $\mu_{i\cdot}$. These comparisons may be pairwise, or more general contrasts, but should be corrected by Tukey's or Scheff'e's technique. 4. Test for significance of Factor B main effects, $H_0 : \tau_j = 0$, for all j versus H_a : not all $\tau_j = 0$. And, again, if we reject H_0, then perform corrected follow-up comparisons of $\mu_{\cdot j}$'s. The above tests for main effects may be conducted using Type 1 sums of squares for balanced experiments, and Type 3 sums of squares (equivalently partial F tests) for unbalanced experiments. For experiments with one or more missing cells, fit a linear model (with extra constraint(s)) and test the main effects using partial F tests.

When interactions may be present, the analysis is more complicated. 1. Fit the model with interactions

$$Y_{ijk} = \mu + \alpha_i + \tau_j + (\alpha\tau)_{ij} + \epsilon_{jik},$$

and test for significance of the model using a partial F test comparing the intercept-only model to the full model. If the hypothesis H_0 : only the intercept is nonzero is rejected, then move to step 2. 2. Test for significance of the interaction terms $H_0 : (\alpha\tau)_{ij} = 0$ for all i, j versus H_a : not all $(\alpha\tau)_{ij} = 0$. 3. If we reject H_0, then perform any desired follow-up tests (CIs) for pairwise comparisons or contrasts of the crossed factor treatment means, the μ_{ij}'s. For multiple such comparisons use Tukey or Scheff'e adjustments. 4. If we fail to reject the null hypothesis of no interaction, then carry out the tests for main effects. The test for Factor A main effects has null and alternative hypotheses $H_0 : \alpha_i = 0$ for all i versus H_a : not all $\alpha_i = 0$. Likewise, the test for Factor B main effects has hypotheses $H_0 : \tau_j = 0$, for all j versus H_a : not all $\tau_j = 0$. 5. For each test of main effects, if we reject the null hypothesis, then we may consider conducting follow-up tests to compare the factor-level mean responses, the $\mu_{i.}$'s and/or $\mu_{.j}$'s. For multiple such comparisons, again, use Tukey or Scheff'e corrections.

Remarks: 1. For the model fit with interaction terms, should we fail to reject the null hypothesis of no interaction, it is generally not appropriate to refit the model without interaction terms. Consider the following reasoning. We used some degrees of freedom (we used data) to estimate the interaction parameters, and we tested and found them to be insignificant. If we then refit the model without those terms, we reuse those degrees of freedom to estimate σ^2; this is an overly optimistic quantification of uncertainty. And, it constitutes a statistical test performed using data conditional on the same data, which often can cause Type 1 error inflation. It is more honest to use the MSE from the full model, with the interaction effects, in all subsequent tests and CIs. 2. When we learn ANOVA we often start with sums of squares. These are especially useful in one-way ANOVA because the derivation of the F test as a comparison of variance estimators under null and alternative hypotheses about the treatment means (the μ_i's) is particularly instructive. However, once we encounter unbalanced data, sums of squares become far less useful. My advice is to always associate ANOVA-type tests with partial F tests, because these always "work", while the sums of squares approach falls apart when we have unbalanced data and/or missing cells. 3. The two-factor model with one (or more) missing (at random) cell(s) is the most troublesome to analyze. There are (at least) three strategies for dealing with the missing cell. One is to simply treat the crossed treatments as a single "treatment" variable and perform one-way ANOVA. The second is to perform "partial-table" analyses. And the third, (and my personal favorite) is to use a multiple linear regression model with an additional constraint for each missing cell. For strategies 2 and 3 we again proceed by testing for interactions, and, if insignificant, testing for main effects. In all three strategies we may perform the appropriate, corrected, follow-up tests.

Let's try out our general strategy using two examples, unbalanced data, and data with a missing cell. The case of balanced data is baseically the same as the unbalanced case.

12.1 Unbalanced Two-Factor Experiment: Chick Weight

The following data comes from an animal breeding experiment where chicks of different breeds are fed different diets and their weights are recorded. The data is unbalanced, but each combination of breeds and feeds is observed.

```
chickgrowth<- read.csv('chick.csv')
chickgrowth$feed<-as.factor(chickgrowth$feed)
chickgrowth$breed<-as.factor(chickgrowth$breed)
chickgrowth
```

```
##     X growth feed breed
## 1   1   3.31    1     1
## 2   2   3.48    1     1
## 3   3   3.36    1     1
## 4   4   3.92    1     2
## 5   5   5.26    1     2
## 6   6   4.72    1     2
## 7   7   5.56    1     3
## 8   8   5.75    1     3
## 9   9   5.62    1     3
## 10 10   6.12    1     4
## 11 11   4.35    2     1
## 12 12   3.69    2     1
## 13 13   5.91    2     2
## 14 14   4.77    2     2
## 15 15   5.22    2     2
## 16 16   5.57    2     3
## 17 17   6.20    2     3
## 18 18   6.03    2     3
## 19 19   7.28    2     4
## 20 20   7.34    2     4
## 21 21   7.77    2     4
## 22 22   4.65    3     1
## 23 23   4.42    3     1
## 24 24   5.39    3     2
## 25 25   5.87    3     2
## 26 26   6.45    3     3
## 27 27   6.32    3     3
## 28 28   7.22    3     4
## 29 29   6.65    3     4
```

We begin our analysis with a model F test. Since the p-value basically zero we reject the null hypothesis that only the intercept term is non-zero. The

interpretation is that at least one of the crossed treatments of breed-by-feed has a mean weight significantly different from the mean weights of the other breed-by-feed treatment groups.

```
full.model <- lm(growth~feed*breed, data = chickgrowth)
SSE.full <- sum(full.model$residuals^2)
int.model <- lm(growth~1, data = chickgrowth)
SSE.int <- sum(int.model$residuals^2)
n <- length(chickgrowth$growth)

F <- ((SSE.int - SSE.full) / 11) / (SSE.full/(n - 12))
1-pf(F,11,n-12)
```

```
## [1] 2.478104e-08
```

Therefore, we consider testing for interaction effects. (I should have mentioned that since we have no reason to suspect there are no interactions we, by default, include interactions in the full model.)

```
no.inter.model <- lm(growth~feed + breed, data = chickgrowth)
SSE.no.inter <- sum(no.inter.model$residuals^2)

F <- ((SSE.no.inter - SSE.full) / 6) / (SSE.full/(n - 12))
1-pf(F,6,n-12)
```

```
## [1] 0.3159945
```

Another way to perform the test is by using type 3 SS. Note we matched the p-value.

```
library(car)
```

```
## Loading required package: carData
```

```
Anova(lm(growth~feed*breed, data = chickgrowth), type = 3)
```

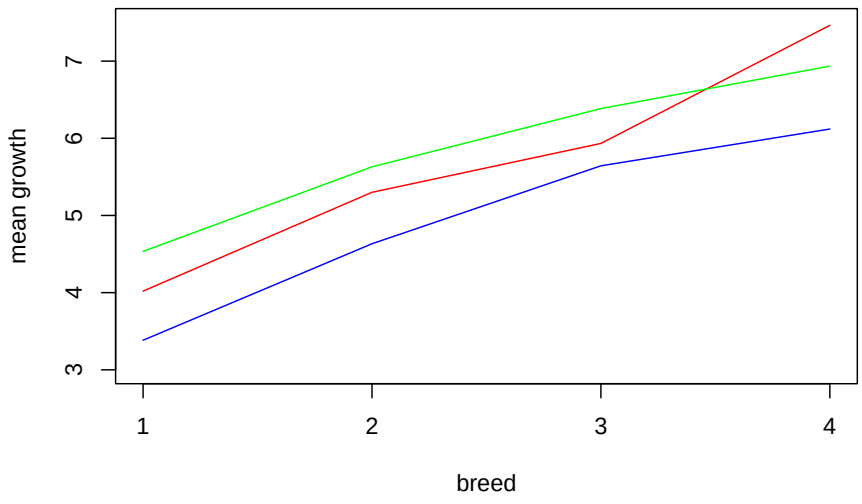
```
## Anova Table (Type III tests)
##
## Response: growth
##             Sum Sq Df  F value    Pr(>F)
## (Intercept) 34.341  1 234.6172 2.220e-11 ***
## feed         1.636  2   5.5881   0.01364 *
```

```
## breed        9.899  3  22.5438 3.693e-06 ***
## feed:breed   1.128  6   1.2842   0.31599
## Residuals    2.488 17
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The test for interactions does not reject the null hypothesis of no interaction. This means the additive model fits the data just fine, and that all relevant comparisons may be performed using factor-level means, rather than crossed treatment means. However, keep hold of the MSE for the full model, as we will use this for subsequent follow-up tests…

The interaction plots below help illustrate why the interaction effects were not significant.

```
means <- aggregate(growth~feed*breed, data = chickgrowth, FUN=mean)
sub1 <- c(1,4,7,10)
sub2 <- c(2,5,8,11)
sub3 <- c(3,6,9,12)
plot(as.numeric(means[sub1,2]), means[sub1,3], type = 'l', xlab = 'breed', col = 'blue', ylab = '
axis(side = 1, at = c(1,2,3,4))
lines(means[sub2,2], means[sub2,3], col = 'red')
lines(means[sub3,2], means[sub3,3], col = 'green')
```



Given we can assume interactions are not present, let's test for main effects… Ope, we already have these from the Type III SS ANOVA results above! Based

on the p-values (≈ 0.014 and 0.000004), both main effects are significant. So, it makes sense to conduct follow-up tests based on factor-level means. I don't have any specific contrasts in mind to test, so I'll simply conduct pairwise comparisons.

Given it seems that breed and feed effects are additive/separable, the experimenters will likely make recommendations about breed and feed based on comparisons between levels of breeds and comparisons between levels of feed. Therefore, Tukey corrections for feed and for breed separately make the most sense. What I'm saying is that there are not likely to be decisions based on breed-feed comparisons jointly, so we can correct for comparing 3 pairs within feed and 6 pairs within breed separately.

Just for brevity, I'll only compute one Tukey-corrected pairwise comparison "by hand". Note, I match the p-value and the LS means computed using the emmeans package. However, I do not match the "t.ratio" in the table initially. The "t.ratio" in the table does not include the $\frac{1}{\sqrt{2}}$ factor in the margin of error necessary for comparing to Tukey quantiles... That's fine, just be careful you know exactly what you're looking at.

Based on the results, if the experimenters are interested in the breed and feed combinations with highest weight (which makes sense) then we see breed 4 has highest weight, and feeds 2 and 3 have higher weight than feed 1, although 2 and 3 are not distinguishable. The conclusion of this experiment is that feed and breed factors largely have additive effects on weight, and that breed 4 and feeds 2 and 3 seem to produce the highest weight chicks.

```
MSE <- (SSE.full/(n - 12))


means <- aggregate(growth~feed*breed, data = chickgrowth, FUN=mean)
LS.means.feed1 <- 0.25*(means[1,3]+means[4,3]+means[7,3]+means[10,3])
n11 <- sum(chickgrowth$feed==1 & chickgrowth$breed==1)
n12 <- sum(chickgrowth$feed==1 & chickgrowth$breed==2)
n13 <- sum(chickgrowth$feed==1 & chickgrowth$breed==3)
n14 <- sum(chickgrowth$feed==1 & chickgrowth$breed==4)
LS.means.feed2 <- 0.25*(means[2,3]+means[5,3]+means[8,3]+means[11,3])
n21 <- sum(chickgrowth$feed==2 & chickgrowth$breed==1)
n22 <- sum(chickgrowth$feed==2 & chickgrowth$breed==2)
n23 <- sum(chickgrowth$feed==2 & chickgrowth$breed==3)
n24 <- sum(chickgrowth$feed==2 & chickgrowth$breed==4)


tukey.feed12 <- (LS.means.feed1-LS.means.feed2) / ((1/sqrt(2))*sqrt(MSE*(1/16)*(1/n11+1

tukey.feed12
```

```
## [1] -5.802434
```

```
1-ptukey(abs(tukey.feed12), 3, n-12)
```

```
## [1] 0.002031653
```

```
t.ratio <- (LS.means.feed1-LS.means.feed2) / (sqrt(MSE*(1/16)*(1/n11+1/n12 + 1/n13 + 1/n14+1/n21+

t.ratio
```

```
## [1] -4.10294
```

```
library(emmeans)
em.feed <- emmeans(lm(growth~feed*breed, data = chickgrowth), c('feed'))
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
em.feed
```

```
##  feed emmean    SE df lower.CL upper.CL
##  1      4.95 0.135 17     4.66     5.23
##  2      5.68 0.117 17     5.43     5.93
##  3      5.87 0.135 17     5.59     6.16
##
## Results are averaged over the levels of: breed
## Confidence level used: 0.95
```

```
pairs(em.feed, adjust = 'tukey')
```

```
##  contrast      estimate    SE df t.ratio p.value
##  feed1 - feed2   -0.734 0.179 17  -4.103  0.0020
##  feed1 - feed3   -0.926 0.191 17  -4.842  0.0004
##  feed2 - feed3   -0.192 0.179 17  -1.073  0.5427
##
## Results are averaged over the levels of: breed
## P value adjustment: tukey method for comparing a family of 3 estimates
```

```
em.breed <- emmeans(lm(growth~feed*breed, data = chickgrowth), c('breed'))
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
em.breed
```

```
##  breed emmean    SE df lower.CL upper.CL
##  1       3.98 0.147 17     3.67     4.29
##  2       5.19 0.138 17     4.90     5.48
##  3       5.99 0.138 17     5.70     6.28
##  4       6.84 0.173 17     6.48     7.20
##
## Results are averaged over the levels of: feed
## Confidence level used: 0.95
```

```
pairs(em.breed, adjust = 'tukey')
```

```
##  contrast        estimate    SE df t.ratio p.value
##  breed1 - breed2   -1.208 0.202 17  -5.993  0.0001
##  breed1 - breed3   -2.008 0.202 17  -9.957  <.0001
##  breed1 - breed4   -2.860 0.227 17 -12.603  <.0001
##  breed2 - breed3   -0.799 0.195 17  -4.104  0.0037
##  breed2 - breed4   -1.652 0.221 17  -7.478  <.0001
##  breed3 - breed4   -0.852 0.221 17  -3.858  0.0063
##
## Results are averaged over the levels of: feed
## P value adjustment: tukey method for comparing a family of 4 estimates
```

12.2 Example: observational study on growth-hormone deficient children with empty cell

From Applied Linear Statistical Models (Fifth Edition) by Kutner et al.: "Synthetic growth hormone was administered at a cI inical research center to growth hormone deficient, short children who had not yet reached pubel1y. The investigator was interested in the effects of a child's gender (Factor A) and bone development (Factor B) on the rate of growth induced by hormone administration. A child's bone development was classified into one of three categories: severely depressed, moderately depressed, mildly depressed. Three children were randomly selected for each gender-bone development group. The response variable (Y) of interest was the difference between the growth rate during growth hormone treatment and the normal growth rate prior to the treatment, expressed in centimeters per month. Four of the 18 children were unable to complete the year-long study, thus creating unequal treatment sample sizes. Note that this is an observational study. All children received the same hormone therapy, and, subsequently, changes in growth rates were observed for children in each bone development-by-gender category. No randomization of treatments to subjects

was employed." We will examine a modified data set where an additional (fifth) response is missing so that we have an empty cell; in particular, the Female-Severely depressed combination is unobserved.

Sex	Bone Development Depression	Mean(n)
Male	Severe	2.0(3)
Male	Moderate	1.9(2)
Male	Mild	0.9(2)
Female	Severe	–
Female	Moderate	2.1(3)
Female	Mild	0.9(3)

```
response <- c(1.4,2.4,2.2,2.1,1.7,0.7,1.1,2.5,1.8,2.0,0.5,0.9,1.3)
bone <- c('severe','severe','severe','moderate','moderate','mild','mild','moderate','moderate','m
sex <- c('male','male','male','male','male','male','male','female','female','female','female','fe
growth <- data.frame(response = response, bone = bone, sex = sex)
```

12.2.1 Linear Model analysis with extra constraint

If we impose sum-to-zero constraints, and remove the interaction corresponding to the missing cell, then we can analyze the available data using one linear model. Since there are a total of 5 cell means, we should have 5 regression parameters, 1 intercept, 2 for bone, 1 for sex, and 1 interaction term. I have chosen to design my own interaction term, which contrasts the mal-female difference in the moderate bone group to the male female difference in the mild bone group. If there is no interaction, these differences should be the same (because they are additive in sex and bone effects). Specifically, the interaction term I am testing is

$$(\mu_{21} - \mu_{22}) - (\mu_{31} - \mu_{32})$$

where the first index i denotes sex ($i = 1$ male and $i = 2$ female) and the second index j denotes bone ($j = 1$ severe, $j = 2$ moderate, $j = 3$ mild).

```
options(contrasts = c('contr.sum', 'contr.sum'))
X.full <- model.matrix(response ~ bone*sex, data = growth)
X.obs <- X.full[,1:4]
X.obs <- cbind(X.obs,c(0,0,0,1,1,-1,-1,-1,-1,1,1,1))
solve(t(X.obs)%*%X.obs)%*%t(X.obs)
```

```
##                          1             2             3          4          5
## (Intercept)  1.111111e-01  1.111111e-01  1.111111e-01  0.04166667  0.04166667
## bone1       -1.111111e-01 -1.111111e-01 -1.111111e-01 -0.04166667 -0.04166667
## bone2       -1.111111e-01 -1.111111e-01 -1.111111e-01  0.20833333  0.20833333
```

```
## sex1         1.387779e-17  1.387779e-17  1.387779e-17 -0.12500000 -0.12500000
##              0.000000e+00  0.000000e+00  0.000000e+00  0.12500000  0.12500000
##                         6             7             8           9          10
## (Intercept)  0.04166667  0.04166667  0.08333333  0.08333333  0.08333333
## bone1        0.20833333  0.20833333 -0.08333333 -0.08333333 -0.08333333
## bone2       -0.04166667 -0.04166667  0.08333333  0.08333333  0.08333333
## sex1        -0.12500000 -0.12500000  0.08333333  0.08333333  0.08333333
##             -0.12500000 -0.12500000 -0.08333333 -0.08333333 -0.08333333
##                        11          12          13
## (Intercept)  0.08333333  0.08333333  0.08333333
## bone1        0.08333333  0.08333333  0.08333333
## bone2       -0.08333333 -0.08333333 -0.08333333
## sex1         0.08333333  0.08333333  0.08333333
##              0.08333333  0.08333333  0.08333333
```

Using this design I can test for interaction using a partial F test. The p-value is 0.711 which implies there is no significant interaction. Be careful about what this test is saying. The interaction term compares responses over sex and mild to moderate bone development disorder. Since we lack data for female-severe, we cannot say whether or not some interaction may occur for that level of bone development disorder.

```
n <- nrow(X.obs)
p <- ncol(X.obs)
hat.beta <- solve(t(X.obs)%*%X.obs)%*%t(X.obs)%*%matrix(response, n,1)
hat.beta
```

```
##              [,1]
## (Intercept)  1.65
## bone1       -0.75
## bone2        0.35
## sex1         0.05
##             -0.05
```

```
SSE.obs <- sum((matrix(response, n,1) - X.obs%*%hat.beta)^2)
X.nointeraction <- X.obs[,1:4]
hat.beta.nointer <- solve(t(X.nointeraction)%*%X.nointeraction)%*%t(X.nointeraction)%*%
SSE.nointer <- sum((matrix(response, n,1) - X.nointeraction%*%hat.beta.nointer)^2)

F <- ((SSE.nointer - SSE.obs)/1)/(SSE.obs/(n-p))
F
```

```
## [1] 0.1476923
```

```
1-pf(F,1,n-p)
```

```
## [1] 0.7107643
```

Given the only testable interaction is not significant, we move on to examining the main effects. The first estimated main effect for bone is given by

$$-1/9Y_1-1/9Y_2-1/9Y_3-1/24Y_4-1/24Y_5+5/24Y_6+5/24Y_7-1/12Y_8-1/12Y_9-1/12Y_10+1/12Y_11+1/12Y_12+1/12Y_13.$$

This is an estimate of the following contrast:

$$-1/3\mu_{11} - 1/12\mu_{21} + 5/12\mu_{31} + 1/4(\mu_{22} - \mu_{32})$$

The difference $\mu_{22} - \mu_{32}$ is the difference in mean response for females in groups moderate and mild. If there is no difference between bone groups (no bone effects) then this difference should be zero because both are average female responses. Likewise, if there are no bone effects, then $-1/3\mu_{11} - 1/12\mu_{21} + 5/12\mu_{31}$ simplifies to a linear combination of $\mu_{.}1$, the average male response; and that linear combination is $-1/3\mu_{.1} - 1/12\mu_{.1} + 5/12\mu_{.1} = 0$.

The partial F test for bone effect is significant; there are significant difference in mean growth between treated and untreated periods depending on one's severity of bone under-development.

```
X.nobone <- X.obs[,-(2:3)]
hat.beta.nobone <- solve(t(X.nobone)%*%X.nobone)%*%t(X.nobone)%*%matrix(response, n,1)
SSE.nobone <- sum((matrix(response, n,1) - X.nobone%*%hat.beta.nobone)^2)

F <- ((SSE.nobone - SSE.obs)/2)/(SSE.obs/(n-p))
F
```

```
## [1] 10.83429
```

```
1-pf(F,1,n-p)
```

```
## [1] 0.01099661
```

There is not a significant difference in mean bone growth between sexes.

```
X.nosex <- X.obs[,-4]
hat.beta.nosex <- solve(t(X.nosex)%*%X.nosex)%*%t(X.nosex)%*%matrix(response, n,1)
SSE.nosex <- sum((matrix(response, n,1) - X.nosex%*%hat.beta.nosex)^2)

F <- ((SSE.nosex - SSE.obs)/1)/(SSE.obs/(n-p))
F
```

```
## [1] 0.1476923
```

```
1-pf(F,1,n-p)
```

```
## [1] 0.7107643
```

Now, which means should we compare? Since only bone is significant we should compare the three bone under-development levels pairwise. However, we should consider whether or not it makes sense to compare the LS mean response for sever bone under-development considering it only contains male responses. In this case, since sex is not significant, I think it is reasonable to make comparisons of severe to moderate and severe to mild. However, if sex was significant, then such comparisons would not make sense as there would be no females to average over within the severe category.' The mild bone underdevelopment group is significantly different from the moderate and severe groups, but the latter two are not distinguishable.

```
MSE <- SSE.obs/(n-p)
LS.bone <- c(2,2,0.9)
ratio.12 <- 0
ratio.13 <- 1.1/sqrt(MSE*(1/2)*(1/3+(1/4)*(1/2+1/3)))
ratio.23 <- 1.1/sqrt(MSE*(1/2)*(1/4)*(1/2+1/3+1/2+1/3))

p12 <- 1-ptukey(ratio.12, 3, n-p)
p12
```

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

```
p13 <- 1-ptukey(ratio.13, 3, n-p)
p13
```

```
## [1] 0.01467942
```

```
p23 <- 1-ptukey(ratio.23, 3, n-p)
p23
```

```
## [1] 0.00720332
```

12.2.2 Partial Table analysis

An alternative strategy is to consider partial table analyses. On table consists of male and female at levels moderate and mild. The second, overlapping table

is all male cells (severe, moderate, mild). In the first, two-by-two table, we may assess interaction. If interaction is present, then we should compare all 5 cell means. If not, we test main effects to determine if male-female comparisons are relevant, and if moderate-mild comparisons are relevant. Using the second, three-by-one table of only male responses, we can compare severe-moderate-mild within males, if desired.

Below, we fit the linear model to the partial table of male-female, moderate-mild responses. The interaction term is given by

$$(\mu_{22} - \mu_{21}) - (\mu_{32} - \mu_{31})$$

which is the mean growth for (female:moderate - male:moderate)-(female:mild - male:mild). If there is no interaction, then the female to male difference is the same at levels moderate and mild, so this contrast of means would be equal to zero. Since the interaction is found to be insignificant, we move on to main effects.

As in the full data analysis above, sex is not significant. I think this fact makes it less clear, rather than more clear, how to proceed. Since sex is not significant and bone under-development is significant, it makes sense to compare mean responses between levels of the bone factor, averaged over sex (LS means). However, now we must choose which table to use. We can compare LS means in the 2×2 partial table, or in the 3×1 table of only male responses, or we can go back to the full data and compare LS means for the bone factor using all the data. The third option matches the full data analysis from above, and makes sense given that sex is not significant. One advantage of the partial table analysis is that if sex is significant, it does not make sense to compare LS means of the bone factor because one of those contains no females; therefore, a partial table analysis seems necessary. In the present case, I would advocate for the full data analysis, comparing LS means of bone considering sex is not significant, in which case the rest of the analysis is the same as above.

```
growth2 <- growth[4:n,]
growth2$bone = factor(growth2$bone, levels = unique(growth2$bone))
library(car)
my.lm<-lm(response~bone*sex, data = growth2)
Anova(my.lm)
```

```
## Anova Table (Type II tests)
##
## Response: response
##           Sum Sq Df F value   Pr(>F)
## bone       3.136  1 25.4270 0.002351 **
## sex        0.024  1  0.1946 0.674571
## bone:sex   0.024  1  0.1946 0.674571
## Residuals  0.740  6
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(my.lm)
```

```
##
## Call:
## lm(formula = response ~ bone * sex, data = growth2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -0.40  -0.20  -0.05   0.20   0.40
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.4500     0.1134  12.793  1.4e-05 ***
## bone1         0.5500     0.1134   4.852  0.00284 **
## sex1          0.0500     0.1134   0.441  0.67457
## bone1:sex1    0.0500     0.1134   0.441  0.67457
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3512 on 6 degrees of freedom
## Multiple R-squared:  0.8114, Adjusted R-squared:  0.7171
## F-statistic: 8.605 on 3 and 6 DF,  p-value: 0.01359
```

```
X <- model.matrix(response~bone*sex, data = growth2)
solve(t(X)%*%X)%*%t(X)
```

```
##                  4      5      6      7          8          9         10
## (Intercept)  0.125  0.125  0.125  0.125 0.08333333 0.08333333 0.08333333
## bone1        0.125  0.125 -0.125 -0.125 0.08333333 0.08333333 0.08333333
## sex1        -0.125 -0.125 -0.125 -0.125 0.08333333 0.08333333 0.08333333
## bone1:sex1  -0.125 -0.125  0.125  0.125 0.08333333 0.08333333 0.08333333
##                      11          12          13
## (Intercept)  0.08333333  0.08333333  0.08333333
## bone1       -0.08333333 -0.08333333 -0.08333333
## sex1         0.08333333  0.08333333  0.08333333
## bone1:sex1  -0.08333333 -0.08333333 -0.08333333
```

Chapter 13

2^k Factorial Experiments

We will wrap up our discussion of ANOVA-type models with 2^k factorial experiments. A study of k binary factors and their effects on a response is a 2^k factorial experiment. The two distinguishing features of these experiments is the fact all factors have only two levels, and the large number of crossed treatments. These result in some unique challenges, but also some simplifications.

13.1 The Model

The ANOVa-type notation becomes cumbersome for 2^k factorial experiments, so we favor the regression-type notation. For example, suppose $k = 3$ so that there are 8 crossed factors. We have main effects, and interactions involving 2, 3, or all 4 factors. To denote the corresponding parameters we use β_j for $j = 1, \ldots, 4$ for main effects, β_{jk} for two-way interactions, and $\beta_{jk\ell}$ for three-way interactions. Then, using the subscript i for the observation $\imath = 1, \ldots, n$, the model can be written

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{3i} + \beta_{12} X_{12i} + \beta_{13} X_{13i} + \beta_{23} X_{23i} + \beta_{123} X_{123i} + \epsilon_i.$$

Note we have 8 regression coefficients corresponding to the 8 crossed treatment means.

For balanced experiments some interesting simplifications emerge. The design matrix may be easily coded using a sum-to-zero format as follows. Let response i in main effects column j be -1 if it has the first level of the factor and +1 if it has the second level. Note that interaction columns of the design are determined by multiplication of the main effects columns. Suppose for $k = 3$ we have 16 responses (2 replicates for each combination of factor levels). The first column of X is all 1's. The second, third, and fourth columns are for main effects, adn the last four are for interactions. The design matrix is as follows:

$$
\begin{array}{cccccccc}
1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\
1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 \\
1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 \\
1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\
1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\
1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & 1 & -1 & 1 & -1 & 1
\end{array}
$$

The design matrix X has the interesting property that any two columns are orthogonal, i.e., their inner (dot) product is zero. Therefore, $X^\top X = 16 I_{16}$ and $(X^\top X)^{-1} = \frac{1}{16} I_{16}$. And, $\hat{\beta} = \frac{1}{16} X^\top Y$, making computation of the estimated regression coefficients very easy. It also means that the estimated regression coefficients are uncorrelated (because $(X^\top X)^{-1}$ has all zeroes on the off-diagonals) with equal variance $\sigma^2/16$ (because the diagonal entries are equal).

13.1.1 Example: Stress Test Study

The following data records the results of a stress test measuring exercise tolerance on male and female adults who smoke (light or heavy) and who have high or low body fat:

```
stress <- c(24.1,29.2,24.6,20.0,21.9,17.6,14.6,15.3,12.3,16.1,9.3,10.8,17.6,18.8,23.2,1
smoking <-c(rep(-1,12), rep(1,12))
fat <- c(rep(-1,6), rep(1,6),rep(-1,6), rep(1,6))
sex <- c(-1,-1,-1,1,1,1,-1,-1,-1,1,1,1,-1,-1,-1,1,1,1,-1,-1,-1,1,1,1)
data.df <- data.frame(stress = stress, smoking = as.factor(smoking), fat=as.factor(fat)

X <- model.matrix(stress~smoking*fat*sex, data = data.df)

t(X)%*%X
```

```
##                   (Intercept) smoking1 fat1 sex1 smoking1:fat1 smoking1:sex1
## (Intercept)                24       12   12   12             6             6
## smoking1                   12       12    6    6             6             6
## fat1                       12        6   12    6             6             3
```

```
## sex1                         12        6    6   12             3             6
## smoking1:fat1                 6        6    6    3             6             3
## smoking1:sex1                 6        6    3    6             3             6
## fat1:sex1                     6        3    6    6             3             3
## smoking1:fat1:sex1            3        3    3    3             3             3
##                     fat1:sex1 smoking1:fat1:sex1
## (Intercept)                 6                 3
## smoking1                    3                 3
## fat1                        6                 3
## sex1                        6                 3
## smoking1:fat1               3                 3
## smoking1:sex1               3                 3
## fat1:sex1                   6                 3
## smoking1:fat1:sex1          3                 3
```

```
solve(t(X)%*%X)%*%t(X)%*%matrix(stress,24,1)
```

```
##                           [,1]
## (Intercept)          25.966667
## smoking1             -6.100000
## fat1                -11.900000
## sex1                 -6.133333
## smoking1:fat1         8.066667
## smoking1:sex1        -1.600000
## fat1:sex1             4.133333
## smoking1:fat1:sex1   -2.233333
```

```
lm(stress~smoking*fat*sex, data = data.df)
```

```
##
## Call:
## lm(formula = stress ~ smoking * fat * sex, data = data.df)
##
## Coefficients:
##        (Intercept)            smoking1                fat1                sex1
##             25.967              -6.100             -11.900              -6.133
##      smoking1:fat1       smoking1:sex1           fat1:sex1  smoking1:fat1:sex1
##              8.067              -1.600               4.133              -2.233
```

```
summary(lm(stress~smoking*fat*sex, data = data.df))
```

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

```
## lm(formula = stress ~ smoking * fat * sex, data = data.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.100 -1.842 -0.950  2.217  4.367
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)          25.967      1.764  14.720 1.01e-10 ***
## smoking1             -6.100      2.495  -2.445 0.026427 *
## fat1                -11.900      2.495  -4.770 0.000209 ***
## sex1                 -6.133      2.495  -2.459 0.025730 *
## smoking1:fat1         8.067      3.528   2.286 0.036198 *
## smoking1:sex1        -1.600      3.528  -0.454 0.656274
## fat1:sex1             4.133      3.528   1.172 0.258526
## smoking1:fat1:sex1   -2.233      4.989  -0.448 0.660434
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.055 on 16 degrees of freedom
## Multiple R-squared:  0.7976, Adjusted R-squared:  0.709
## F-statistic: 9.007 on 7 and 16 DF,  p-value: 0.0001525
```

13.2 Unreplicated Factorial Experiments

Often, in practice, k is very large, so that not enough sample size is available to run a replicated experiment. When only 1 response is available for each crossed treatment (i.e. $n = 2^k$), the 2^k factorial experiment is called "unreplicated", and no degrees of freedom are available to estimate the variance σ^2.

In unreplicated experiments, there are a few strategies available for recovering an estimate of σ^2. 1. We can simply assume some interactions (usually higher-order first) are zero, and then the degrees of freedom associated with these may be put towards estimating σ^2. 2. Graphical methods (Pareto plots). 3. Center-point replication.

The Pareto plot is a graphical display showing the ratio of effect sum of squares to total sum of squares where effect sum of squares is $n\widehat{\beta}_j^2$ for each of the 2^k coefficients. A common technique is to use the effects with smallest sums of squares to pool to obtain a variance estimate. The resulting variance estimate is likely to be an underestimate.

Center-point replication is sometimes implementable for factors based on underlying continuous variables. For example, suppose a factor is "level of watering" of a plant with levels low and high corresponding to amount of 0.5 and 1 liters. The center-point treatment is 0.75 liters. For estimation of the variance, we have

to include replicates with center-point factor levels. The "pure-error" estimate $\hat{\sigma}^2$ is given by the sample variance of the responses at center-point treatment levels.

13.2.1 Example: Granola bar experiment

The following data is from Pecos Foods Company in a 2^4 unreplicated factorial experiment to assess the effects of processing temperature, preservative, moisture, and acidity on microbial growth in granola bars they produce for human consumption.

```
options(contrasts = c('contr.sum', 'contr.sum'))
response <- c(5.55,4.47,5.19,4.32,10.54,11.56,5.08,5.45,5.12,5.63,6.18,5.24,10.73,10.33,6.53,4.93
temp <- c(1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1)
preservative <- c(1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1)
moisture <- c(1,1,1,1,-1,-1,-1,-1,1,1,1,1,-1,-1,-1,-1)
acidity <- c(1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1)
granola <- data.frame(response=response, temp=as.factor(temp), preservative = as.factor(preservat
my.lm <- lm(response~preservative*moisture*acidity*temp, data = granola)
summary(my.lm)
```

```
##
## Call:
## lm(formula = response ~ preservative * moisture * acidity * temp,
##     data = granola)
##
## Residuals:
## ALL 16 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##                                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)                              6.67812        NaN     NaN      NaN
## preservative1                           -1.31312        NaN     NaN      NaN
## moisture1                                1.46563        NaN     NaN      NaN
## acidity1                                 0.15813        NaN     NaN      NaN
## temp1                                   -0.18687        NaN     NaN      NaN
## preservative1:moisture1                 -1.33313        NaN     NaN      NaN
## preservative1:acidity1                   0.19687        NaN     NaN      NaN
## moisture1:acidity1                      -0.17188        NaN     NaN      NaN
## preservative1:temp1                     -0.19313        NaN     NaN      NaN
## moisture1:temp1                          0.11063        NaN     NaN      NaN
## acidity1:temp1                          -0.11688        NaN     NaN      NaN
## preservative1:moisture1:acidity1         0.04938        NaN     NaN      NaN
## preservative1:moisture1:temp1           -0.03812        NaN     NaN      NaN
```

```
## preservative1:acidity1:temp1           -0.13812        NaN     NaN      NaN
## moisture1:acidity1:temp1               -0.30688        NaN     NaN      NaN
## preservative1:moisture1:acidity1:temp1  0.06937        NaN     NaN      NaN
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 15 and 0 DF,  p-value: NA
```

One way to estimate the variance is to assume higher-order interactions are zero. Let's assume the three-way and four-way interactions are zero. That gives us 5 df with which to estimate the variance.

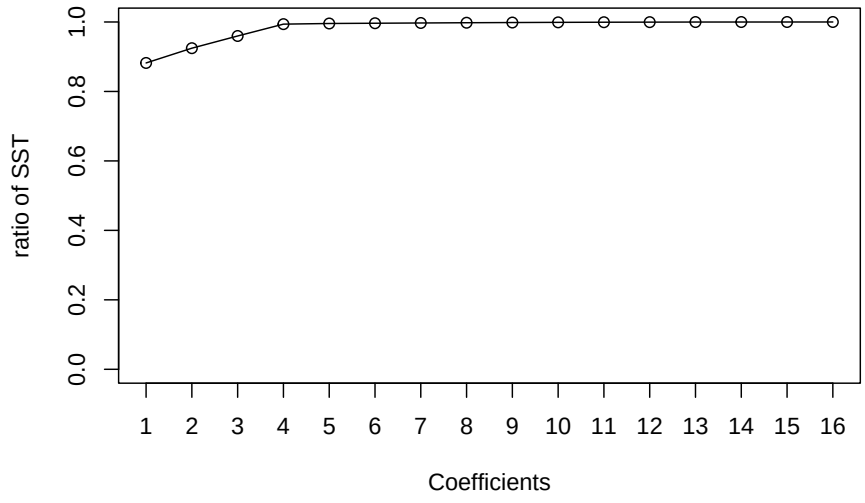
```
options(contrasts = c('contr.sum', 'contr.sum'))
my.lm <- lm(response~preservative*moisture + preservative*acidity + preservative*temp +
summary(my.lm)
```

```
##
## Call:
## lm(formula = response ~ preservative * moisture + preservative *
##     acidity + preservative * temp + moisture * acidity + moisture *
##     temp + acidity * temp, data = granola)
##
## Residuals:
##        1        2        3        4        5        6        7        8
##  0.50312 -0.60187  0.11063 -0.01188 -0.22687  0.32563 -0.38688  0.28812
##        9       10       11       12       13       14       15       16
## -0.42687  0.52563 -0.18688  0.08813  0.15062 -0.24938  0.46313 -0.36437
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                6.6781     0.1562  42.760 1.32e-07 ***
## preservative1             -1.3131     0.1562  -8.408 0.000390 ***
## moisture1                  1.4656     0.1562   9.384 0.000232 ***
## acidity1                   0.1581     0.1562   1.012 0.357769
## temp1                     -0.1869     0.1562  -1.197 0.285114
## preservative1:moisture1   -1.3331     0.1562  -8.536 0.000363 ***
## preservative1:acidity1     0.1969     0.1562   1.261 0.263080
## preservative1:temp1       -0.1931     0.1562  -1.237 0.271156
## moisture1:acidity1        -0.1719     0.1562  -1.101 0.321245
## moisture1:temp1            0.1106     0.1562   0.708 0.510384
## acidity1:temp1            -0.1169     0.1562  -0.748 0.487933
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6247 on 5 degrees of freedom
```

```
## Multiple R-squared:  0.9795, Adjusted R-squared:  0.9386
## F-statistic: 23.95 on 10 and 5 DF,  p-value: 0.001323
```

Let's construct a Pareto chart. Based on the chart, we might only include preservative, preservative:moisture, and the intercept in the model, and use the remaining 13 df to estimate the variance. But, be careful as this is a data-dependent decision.

```
options(contrasts = c('contr.sum', 'contr.sum'))
my.lm <- lm(response~preservative*moisture*acidity*temp, data = granola)
SST <- sum((response - mean(response))^2)
beta.ss.ratio <-(16*(my.lm$coefficients)^2)/SST
terms <- c('intercept', attributes(my.lm$terms)$term.labels)
results <- cbind(terms, as.numeric(beta.ss.ratio))
results <- results[order(results[,2], decreasing = TRUE),]
csum <- cumsum(results[,2])
plot(1:16, csum/csum[16], type = 'o', xlab = 'Coefficients', ylab = 'ratio of SST', xaxt = 'n', y
axis(1, at = 1:16)
```




```
results[,1]
```

```
##  [1] "intercept"                          "moisture"
##  [3] "preservative:moisture"              "preservative"
##  [5] "moisture:acidity:temp"              "preservative:acidity"
```

```
##  [7] "preservative:temp"                 "temp"
##  [9] "moisture:acidity"                  "acidity"
## [11] "preservative:acidity:temp"         "acidity:temp"
## [13] "moisture:temp"                     "preservative:moisture:acidity:temp"
## [15] "preservative:moisture:acidity"     "preservative:moisture:temp"
```

```
n <- length(granola$response)

X <- model.matrix(response~moisture+preservative + preservative*moisture, data = granol
X.r <- X[,-2]
hat.beta <- solve(t(X.r)%*%X.r)%*%t(X.r)%*%matrix(granola$response,n,1)

hat.sigma2 <- sum((granola$response - X.r %*%hat.beta)^2)/(n-ncol(X.r))
hat.sigma2
```

```
## [1] 3.029456
```

Suppose four center-point replicates are 7.23, 7.89, 7.801, 7.39. Then, the variance estimate is the sample variance of these responses, or, $\hat{\sigma}^2 = 0.101$. We may use $\hat{\sigma}^2$ in t-tests of significance of the fitted regression coefficients.

```
options(contrasts = c('contr.sum', 'contr.sum'))
response <- c(5.55,4.47,5.19,4.32,10.54,11.56,5.08,5.45,5.12,5.63,6.18,5.24,10.73,10.33
temp <- c(1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1,1,-1)
preservative <- c(1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1)
moisture <- c(1,1,1,1,-1,-1,-1,-1,1,1,1,1,-1,-1,-1,-1)
acidity <- c(1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1)
granola <- data.frame(response=response, temp=as.factor(temp), preservative = as.factor
my.lm <- lm(response~preservative*moisture*acidity*temp, data = granola)

round(2*(1-pt(abs(my.lm$coefficients)/sqrt(0.101),3)),4)
```

```
##                             (Intercept)                             preservative1
##                                  0.0002                                    0.0257
##                               moisture1                                 acidity1
##                                  0.0192                                    0.6530
##                                   temp1                    preservative1:moisture1
##                                  0.5979                                    0.0247
##                     preservative1:acidity1                       moisture1:acidity1
##                                  0.5795                                    0.6262
##                        preservative1:temp1                          moisture1:temp1
##                                  0.5863                                    0.7508
##                           acidity1:temp1          preservative1:moisture1:acidity1
##                                  0.7375                                    0.8864
```

```
##          preservative1:moisture1:temp1           preservative1:acidity1:temp1
##                                 0.9121                                 0.6932
##              moisture1:acidity1:temp1 preservative1:moisture1:acidity1:temp1
##                                 0.4055                                 0.8412
```

Chapter 14

Linear Regression

In this chapter we study multiple linear regression under the Gauss-Markov model

$$Y = X\beta + \epsilon$$

where Y is an $n \times 1$ vector of responses, X is and $n \times (p+1)$ design matrix containing a leading $n \times 1$ vector of ones (for an intercept term) and n observations on p continuous covariates (also called independent, predictor, or explanatory variables). Recall the model assumes a linear relationship between the (conditional) mean of the response and the covariates , i.e., $E(Y|X) = X\beta$, and normal random residuals with constant variance, $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$.

Throughout this chapter we will use a large example data set to illustrate several concepts related to multiple linear regression models. The data set is the *diamonds* dataframe available in the *ggplot2* package in R. The data set contains prices of over 50000 diamonds, which we will predict using the size (carat) and quality (cut, color, and clarity) of the diamonds.

Among the concepts covered in this section are the following: - Inference on the regression coefficients and conditional means (tests and CIs) - Prediction intervals for a new response (the price of a new diamond) - Interpretation of regression coefficients - Choosing a model - Diagnostics and residuals - Dealing with outliers and influential observations

14.1 Diamonds dataset

The R package ggplot2 includes the dataframe diamonds.

```
library(ggplot2)
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
## 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

```
levels(diamonds$cut)
```

```
## [1] "Fair"      "Good"      "Very Good" "Premium"   "Ideal"
```

```
levels(diamonds$clarity)
```

```
## [1] "I1"   "SI2"  "SI1"  "VS2"  "VS1"  "VVS2" "VVS1" "IF"
```

```
levels(diamonds$color)
```

```
## [1] "D" "E" "F" "G" "H" "I" "J"
```

Cut, color, and clarity are coded as ordinal categorical variables. It may be helpful to use numeric versions of these; for example, cut cut.num coded as an integer 1, 2, 3, 4, or 5, rather than an ordinal variable.

```
diamonds$color.num <- as.numeric(diamonds$color)
diamonds$cut.num <- as.numeric(diamonds$cut)
diamonds$clarity.num <- as.numeric(diamonds$clarity)
```

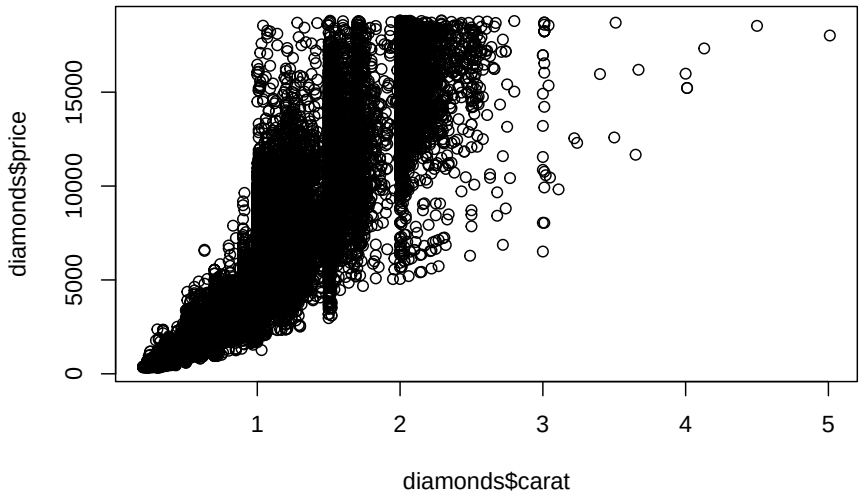
The model we build may depend on what purpose it is to be used for. - Inference: models built to be interpretable and offer best explanation of response in terms of explanatory variables - Prediction: models built to predict the response given explanatory variables Models built for inference tend to be simpler/include fewer variables than models built for prediction. For example, a model built for inference may exclude explanatory variables exhibiting collinearity, because collinearity makes the model more difficult to interpret, but collinearity generally does not make the model worse at prediction.
First we will build a model for inference. Since we are interested in hypothesis testing as part of inference, we will check all the model assumptions. These are less important for prediction.

14.2 Checking Linearity

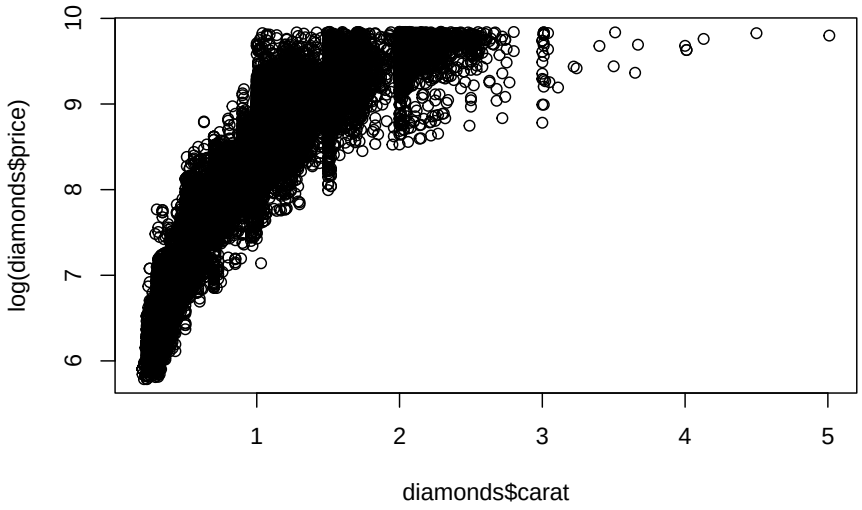
Our multiple linear regression model assumes the conditional mean of response (price) given explanatory variables(carat, cut, color, clarity) is a linear function of those variables. For a given model we can check linearity by examining residuals versus predicted values; there should be no pattern in that plot. We can also do a preliminary check by plotting the response versus each explanatory variable (so long as there are not too many) to see whether any variables exhibit a nonlinear relationship with response. If price is non-linear in a variable, then we can use a transformation to make the relationship more linear.

The square root transformation on price works best in the plot of carat versus price.

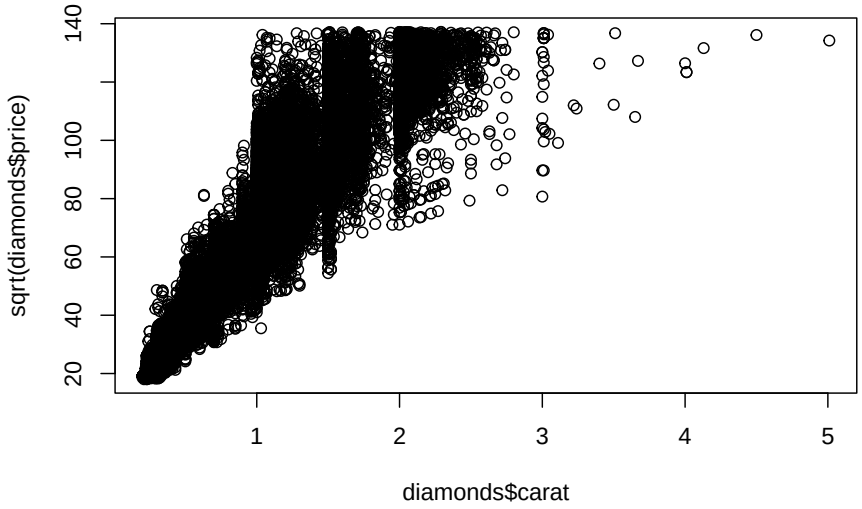
```
plot(diamonds$carat, diamonds$price)
```



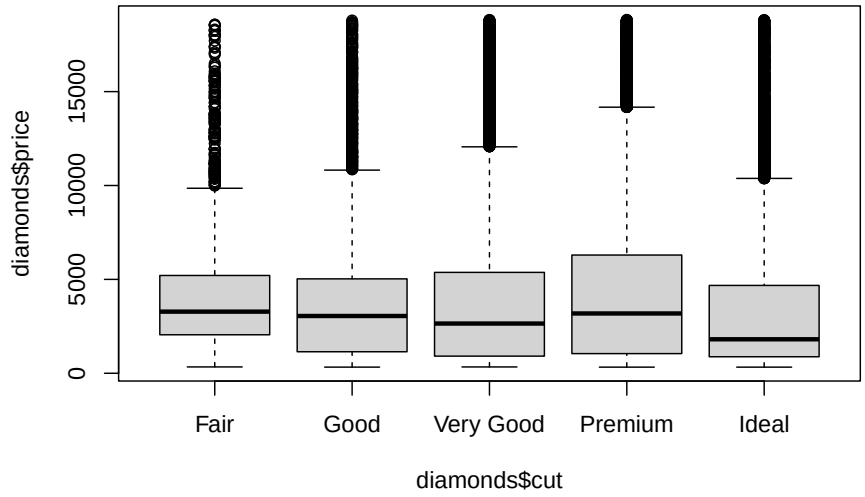

```
plot(diamonds$carat, log(diamonds$price))
```



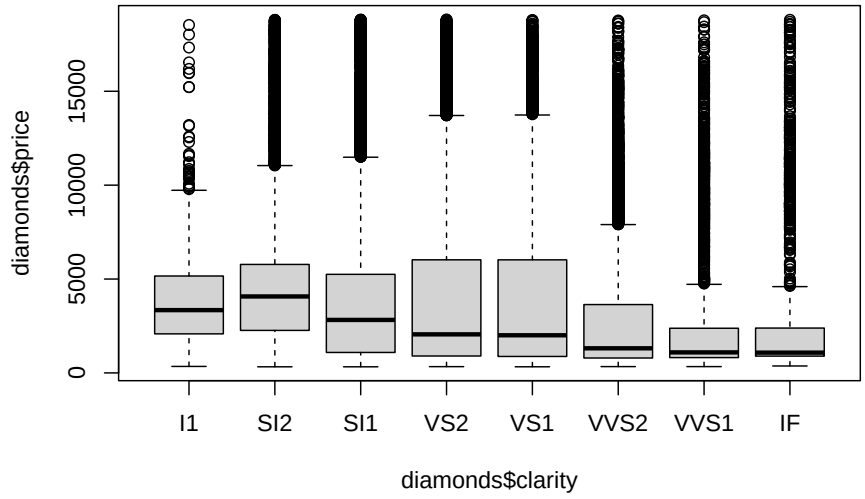

```
plot(diamonds$carat, sqrt(diamonds$price))
```



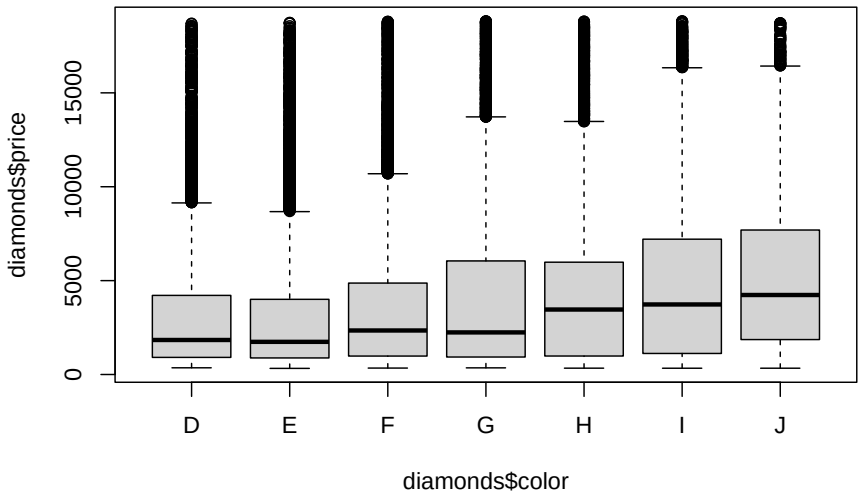
```
boxplot(diamonds$price~diamonds$cut)
```




```
boxplot(diamonds$price~diamonds$clarity)
```

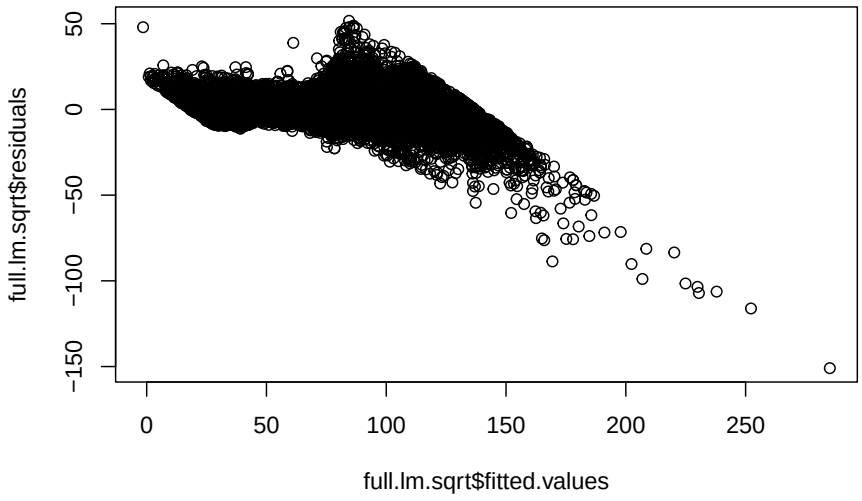


```
boxplot(diamonds$price~diamonds$color)
```



Note that if we fit a model in R with the ordinal-factor versions of the diamond quality variables R uses as many orthogonal polynomial contrasts as it can for each ordinal factor.

```
full.lm.sqrt <- lm(sqrt(price)~carat+cut+color+clarity, data = diamonds)
plot(full.lm.sqrt$fitted.values,full.lm.sqrt$residuals)
```




```
summary(full.lm.sqrt)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut + color + clarity, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -150.936   -3.296   -0.478    2.824   51.690
##
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept)   0.56869    0.07737    7.351 2.00e-13 ***
## carat        64.55472    0.06659  969.371  < 2e-16 ***
## cut.L         3.50254    0.11253   31.124  < 2e-16 ***
## cut.Q        -1.69032    0.09912  -17.054  < 2e-16 ***
## cut.C         1.10438    0.08609   12.828  < 2e-16 ***
## cut^4         0.02937    0.06894    0.426    0.670
## color.L     -13.98189    0.09802 -142.646  < 2e-16 ***
## color.Q      -4.61348    0.08921  -51.712  < 2e-16 ***
## color.C      -0.79061    0.08340   -9.480  < 2e-16 ***
## color^4       0.67221    0.07659    8.777  < 2e-16 ***
## color^5      -0.54516    0.07236   -7.533 5.02e-14 ***
## color^6      -0.09971    0.06579   -1.516    0.130
```

```
## clarity.L    26.37314    0.17062  154.572  < 2e-16 ***
## clarity.Q   -12.45929    0.15953  -78.100  < 2e-16 ***
## clarity.C     6.50359    0.13658   47.619  < 2e-16 ***
## clarity^4    -2.37701    0.10924  -21.761  < 2e-16 ***
## clarity^5     1.60112    0.08915   17.960  < 2e-16 ***
## clarity^6     0.05858    0.07768    0.754    0.451
## clarity^7     0.49801    0.06853    7.267 3.72e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.402 on 53921 degrees of freedom
## Multiple R-squared:  0.9501, Adjusted R-squared:   0.95
## F-statistic: 5.7e+04 on 18 and 53921 DF,  p-value: < 2.2e-16
```

```
AIC(full.lm.sqrt)
```

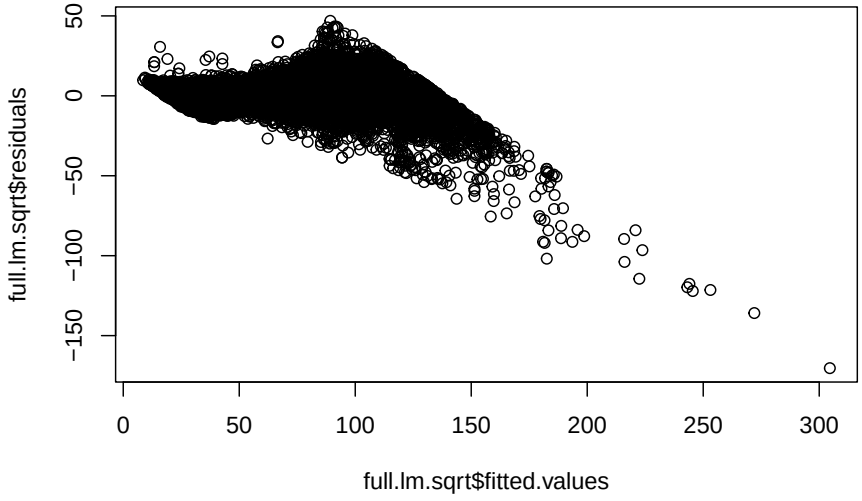
```
## [1] 353387.4
```

```
BIC(full.lm.sqrt)
```

```
## [1] 353565.3
```

A simpler model re-codes the ordinal factors as integers. This model is equivalent to the above model with only linear contrasts for each factor, hence the large difference in number of parameters.

```
full.lm.sqrt <- lm(sqrt(price)~carat+cut.num+color.num+clarity.num, data = diamonds)
plot(full.lm.sqrt$fitted.values,full.lm.sqrt$residuals)
```




```
summary(full.lm.sqrt)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut.num + color.num + clarity.num,
##     data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -170.349   -3.303   -0.046    3.146   46.910
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.80669    0.15774  -17.79   <2e-16 ***
## carat       63.83681    0.07256  879.73   <2e-16 ***
## cut.num      0.79943    0.02781   28.75   <2e-16 ***
## color.num   -2.34832    0.01888 -124.38   <2e-16 ***
## clarity.num  3.20315    0.02017  158.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.061 on 53935 degrees of freedom
## Multiple R-squared:  0.9392, Adjusted R-squared:  0.9392
## F-statistic: 2.084e+05 on 4 and 53935 DF,  p-value: < 2.2e-16
```

```
AIC(full.lm.sqrt)
```

```
## [1] 363946.8
```

```
BIC(full.lm.sqrt)
```

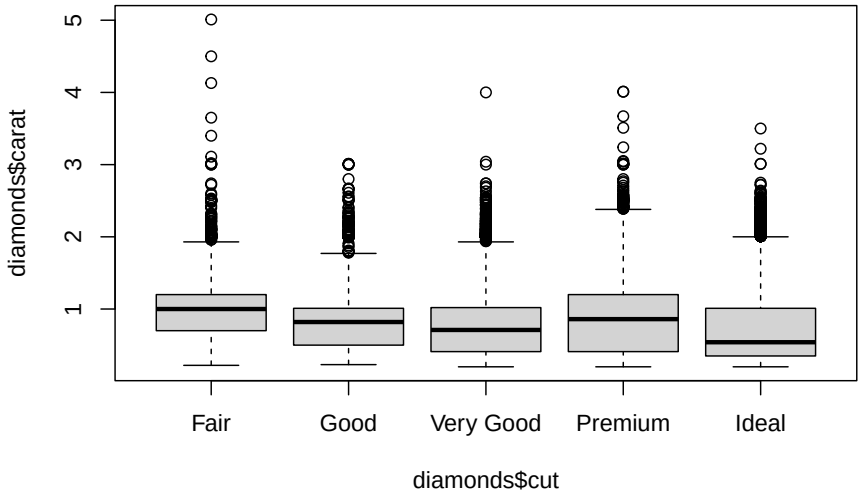
```
## [1] 364000.1
```

Our residuals have a pattern. We may be able to use "interactions" to obtain a better-fitting model due to *multicollinearity*—another term for correlation between covariates.

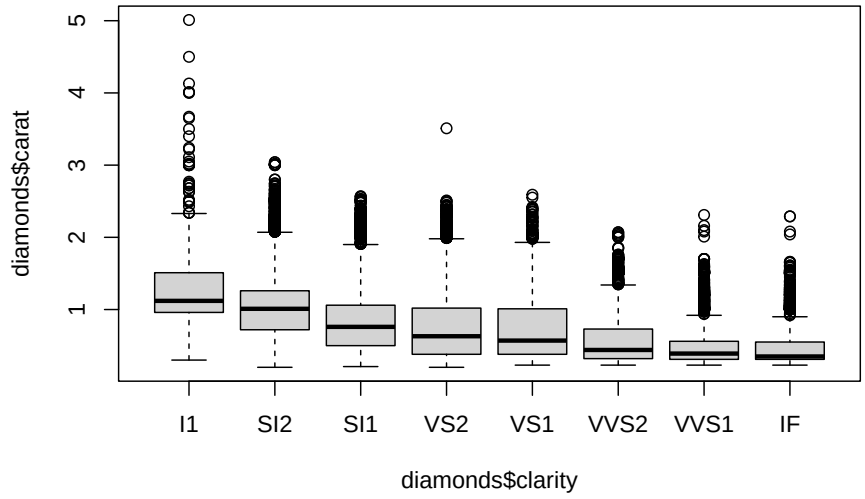
14.3 Multicollinearity

We see that diamonds of the worst cuts, colors, and clarities are also the largest diamonds on average. That makes sense. These diamonds are not worth much of anything unless they are large; the small diamonds of poor quality didn't even make it to the market for diamonds. There is weak multicollinearity that probably is not enough to harm the interpretability of the model if all variables are included.

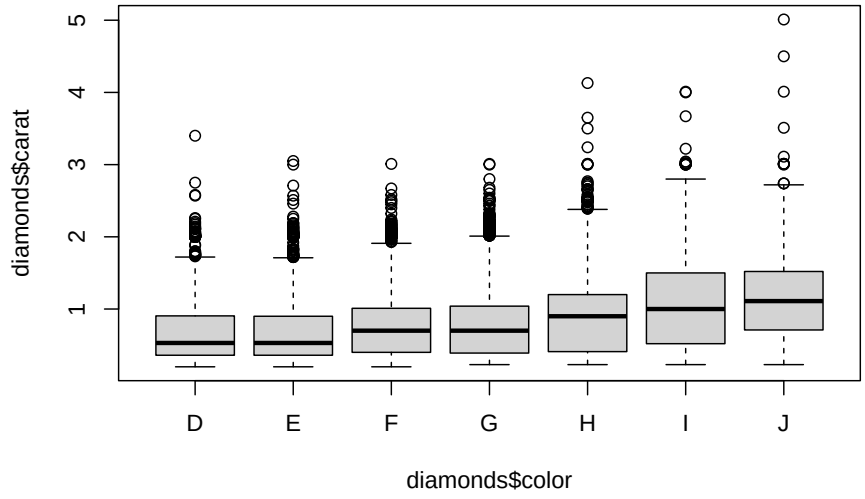
```
boxplot(diamonds$carat~diamonds$cut)
```



```
boxplot(diamonds$carat~diamonds$clarity)
```




```
boxplot(diamonds$carat~diamonds$color)
```



```
cor(diamonds$carat,as.numeric(diamonds$cut))
```

```
## [1] -0.134967
```

```
cor(diamonds$carat,as.numeric(diamonds$clarity))
```

```
## [1] -0.3528406
```

```
cor(diamonds$carat,as.numeric(diamonds$color))
```

```
## [1] 0.2914368
```

14.4 Model with carat interactions

Including the carat interaction with cut, color, and clarity substantially improves the fit of the model as measured by R^2 and R^2_{adj}, the residual error/variance, and as evidenced by the residual vs predicted plot.

```
full.lm.sqrt <- lm(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num
summary(full.lm.sqrt)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut.num + color.num + clarity.num +
##     carat * clarity.num + carat * color.num + carat * cut.num,
##     data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -77.115  -2.174  -0.087   2.314  34.034
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        5.35572    0.19406   27.60  < 2e-16 ***
## carat             52.84947    0.20639  256.06  < 2e-16 ***
## cut.num           -0.23567    0.03783   -6.23 4.71e-10 ***
## color.num          0.45156    0.02413   18.71  < 2e-16 ***
## clarity.num       -0.69980    0.02430  -28.80  < 2e-16 ***
## carat:clarity.num  5.63633    0.02855  197.40  < 2e-16 ***
## carat:color.num   -3.39270    0.02518 -134.74  < 2e-16 ***
## carat:cut.num      1.39771    0.03972   35.19  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.771 on 53932 degrees of freedom
## Multiple R-squared:  0.9723, Adjusted R-squared:  0.9723
## F-statistic: 2.701e+05 on 7 and 53932 DF,  p-value: < 2.2e-16
```

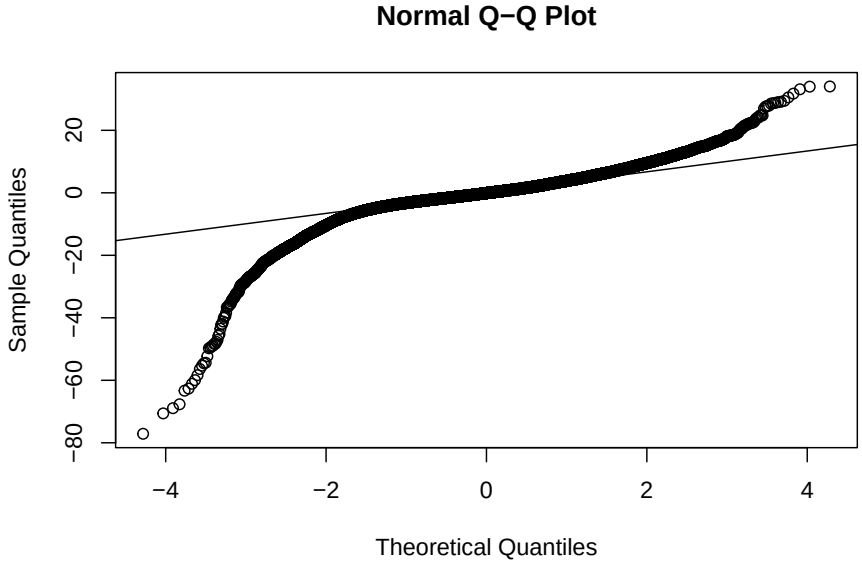
```
AIC(full.lm.sqrt)
```

```
## [1] 321649.3
```

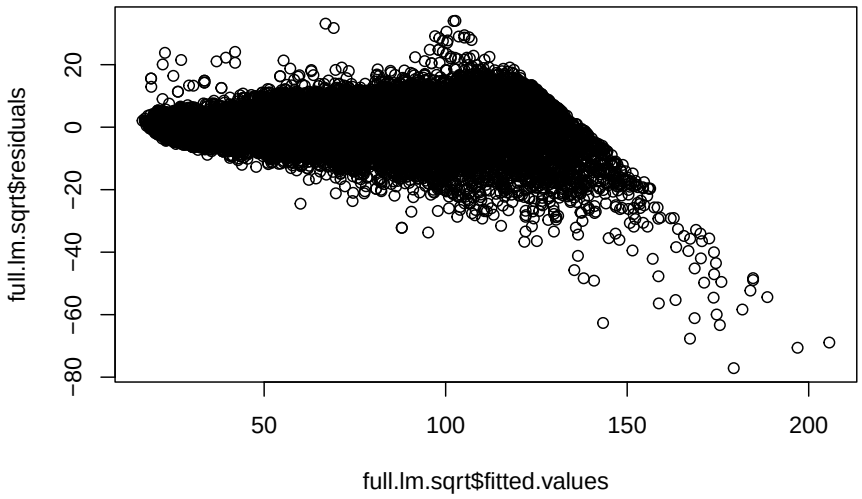
```
BIC(full.lm.sqrt)
```

```
## [1] 321729.4
```

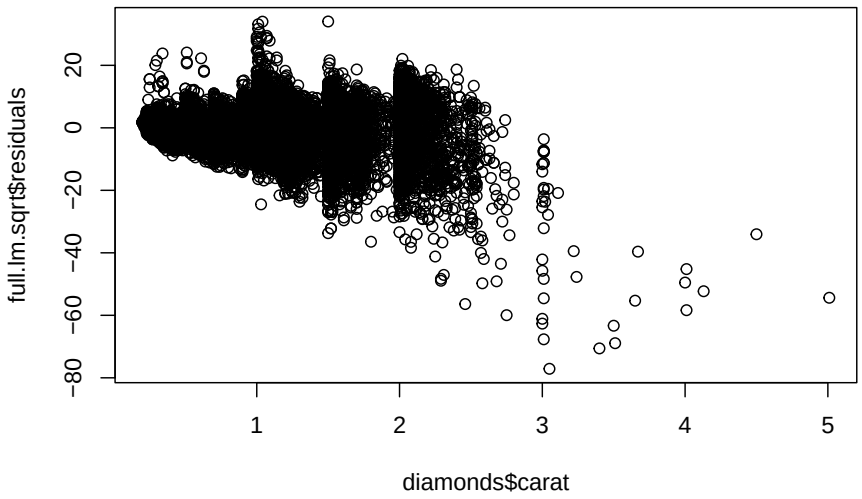
```
qqnorm(full.lm.sqrt$residuals)
qqline(full.lm.sqrt$residuals)
```




```
plot(full.lm.sqrt$fitted.values,full.lm.sqrt$residuals)
```




```
plot(diamonds$carat,full.lm.sqrt$residuals)
```



14.5 Interpreting the model

What about interpretability? We need to be specific about our interpretations, but the (simple) interactions model is certainly interpretable. For example, the first diamond in the data has carat 0.23, it is cut, color, and clarity levels 5, 2, and 2. The interpretation of the fitted betas is as follows, if we hold cut, color, and clarity constant, and increase carat by , say, 0.05, then we will increase sqrt(price) by $3.216264 = 0.05*(52.84947+5*1.39771-2*3.39270+2*5.63633)$. For another example, suppose if that diamond were cut level 4 instead of 5? Then, it's sqrt(price) would change by $-0.0858033 = 0.23567 - 0.23 * 1.39771$.

```
as.matrix(head(diamonds))[1,]
```

```
##       carat         cut       color     clarity       depth       table
##      "0.23"     "Ideal"         "E"       "SI2"      "61.5"        "55"
##       price           x           y           z   color.num     cut.num
##       "326"      "3.95"      "3.98"      "2.43"         "2"         "5"
## clarity.num
##         "2"
```

```
full.lm.sqrt$fitted.values[1]
```

```
##        1
## 18.47571
```

```
X<- model.matrix(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num + carat*colo
X[1,]
```

```
##       (Intercept)             carat           cut.num         color.num
##              1.00              0.23              5.00              2.00
##       clarity.num carat:clarity.num   carat:color.num     carat:cut.num
##              2.00              0.46              0.46              1.15
```

14.6 Checking Constant Variance

As you may have noticed, the observed residuals "fan out"; they increase in magnitude with the value of the predicted sqrt(price). What does this mean for our model/inferences? One consequence is that our tests/CIs for conditional means may not be valid in terms of their nominal α Type 1 error/coverage probability for explanatory variable values with a large estimated mean sqrt(price) value. The variance of the fitted conditional mean $\widehat{\mu}_{Y|x} = x^\top \widehat{\beta}$ is larger than

estimated by $MSEx^\top(X^\top X)^{-1}x$ for large values of $x^\top\hat{\beta}$. The reason is that the variance of ϵ_i seems to be $\sigma^2 \times$ carat rather than σ^2.

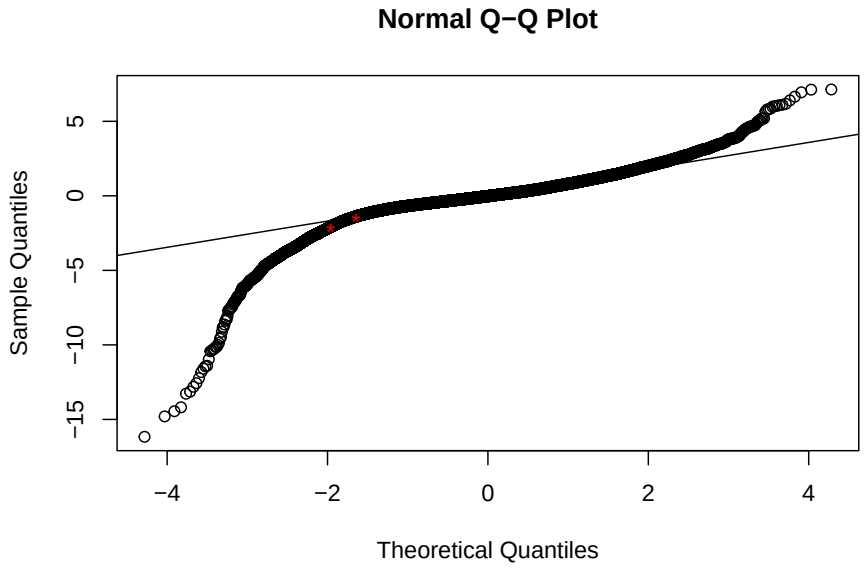
We will come back to this topic later to study what can be done about non-constant variance.

14.7 Normality

How bad is this? The residuals are "normal" for about the middle 90-95 % of their distribution. There are some very extreme residuals; these are diamonds with prices poorly predicted by the model. We should be cautious when interpreting the model for these diamonds. We have likely missed some important information about their prices.

```
qqnorm(full.lm.sqrt$residuals/sd(full.lm.sqrt$residuals))
qqline(full.lm.sqrt$residuals/sd(full.lm.sqrt$residuals), probs = c(0.05, 0.95))

sorted <- sort(full.lm.sqrt$residuals)
points(-1.644854, -1.381829, col = 'red', pch = '*')
points(-1.96, -2.067442 , col = 'red', pch = '*')
```



14.8 Addressing non-constant variance using WLS

Our increasing variance in residuals as a function of predicted price seems to have to do with carat. We can perform a weighted least squares estimation by weighting the observations by carat. The idea is that prices of diamonds with larger carat size have more variance, so we should downweight these observations compared to observations of smaller carat diamonds.
In R we can do this by using the lm() function with the wt option equal to 1/carat. Using weights does not "correct" the non-constant variance; rather, we are incorporating the non-constant variance into the model.

```
get.weights <- lm(full.lm.sqrt$residuals^2 ~ full.lm.sqrt$fitted.values)


wls <- lm(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num + carat*color.num +
summary(wls)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut.num + color.num + clarity.num +
##     carat * clarity.num + carat * color.num + carat * cut.num,
##     data = diamonds, weights = 1/diamonds$carat)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -45.644  -2.700  -0.031   2.874  42.921
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)        3.82491    0.13972   27.376  < 2e-16 ***
## carat             54.59492    0.19776  276.068  < 2e-16 ***
## cut.num           -0.08956    0.02699   -3.318 0.000907 ***
## color.num          0.15453    0.01704    9.067  < 2e-16 ***
## clarity.num       -0.32964    0.01717  -19.199  < 2e-16 ***
## carat:clarity.num  5.21950    0.02755  189.471  < 2e-16 ***
## carat:color.num   -3.11506    0.02404 -129.588  < 2e-16 ***
## carat:cut.num      1.23565    0.03821   32.339  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.583 on 53932 degrees of freedom
## Multiple R-squared:  0.9764, Adjusted R-squared:  0.9764
## F-statistic: 3.192e+05 on 7 and 53932 DF,  p-value: < 2.2e-16
```

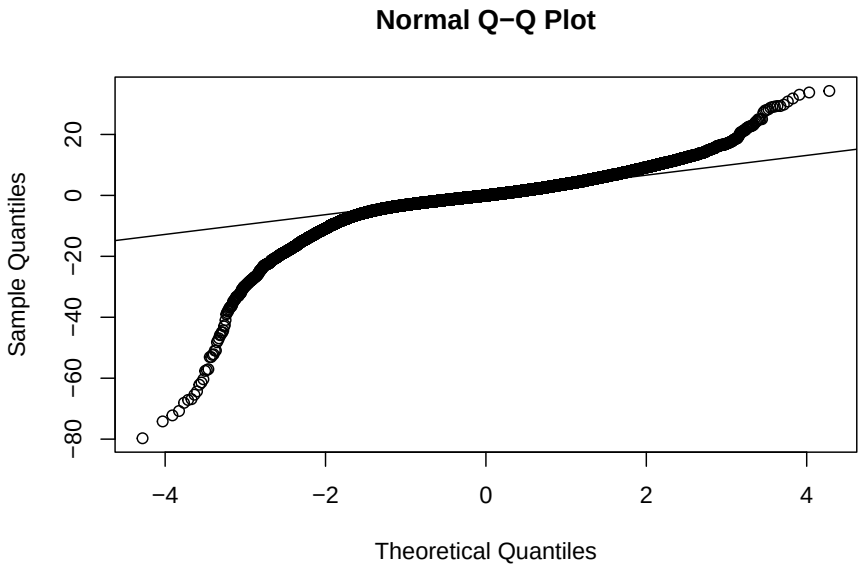
```
AIC(wls)
```

```
## [1] 296006.8
```

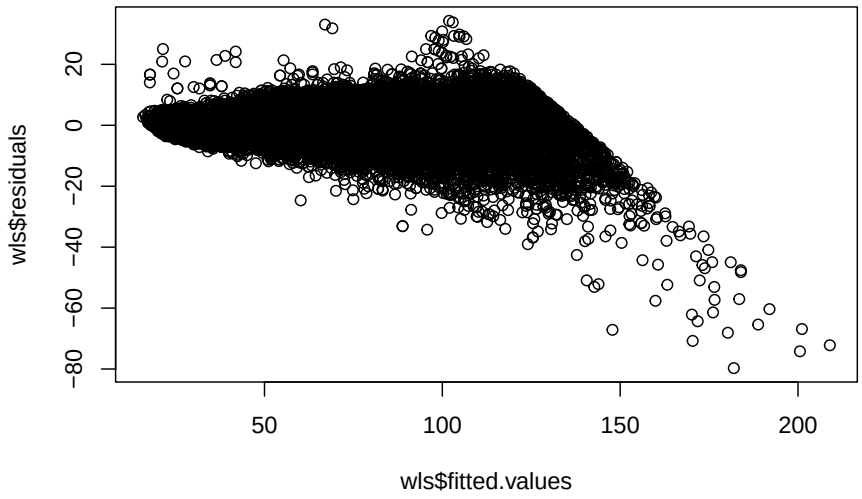
```
BIC(wls)
```

```
## [1] 296086.8
```

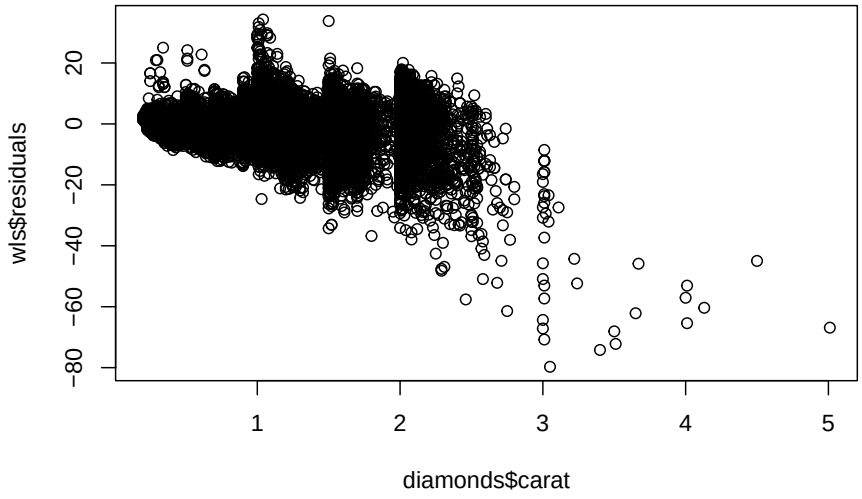
```
qqnorm(wls$residuals)
qqline(wls$residuals)
```




```
plot(wls$fitted.values,wls$residuals)
```




```
plot(diamonds$carat,wls$residuals)
```



14.9 Inferences using WLS

Let's compare confidence/prediction intervals using the WLS and OLS models. The WLS inferences are more "honest" than the OLS inferences because the OLS model's assumption of constant variance is clearly violated. But, let's see what practical difference it makes. 1. 95% CI for the carat beta 2. 95% CI for mean price for carat = 1.5, cut = 3, clarity = 5, color = 2. This is a fairly large diamond of good quality. 3. 95% Prediction interval for the same.

```
X <- model.matrix(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num +
Y <- sqrt(diamonds$price)
n <- nrow(X)
p <- ncol(X)
W <- (1/diamonds$carat)

### OLS inferences
XX.inv <- solve(t(X)%*%X)
beta.hat.ols <- XX.inv%*%t(X)%*%Y
beta.hat.ols
```

```
##                          [,1]
## (Intercept)         5.3557161
## carat              52.8494746
## cut.num            -0.2356664
## color.num           0.4515624
## clarity.num        -0.6998048
## carat:clarity.num   5.6363319
## carat:color.num    -3.3927023
## carat:cut.num       1.3977070
```

```
MSE.ols <- (1/(n-p))*sum((Y - X%*%beta.hat.ols)^2)
MSE.ols
```

```
## [1] 22.76062
```

```
# CI for carat beta
c(beta.hat.ols[2] - qt(0.975, n-p)*sqrt(MSE.ols*XX.inv[2,2]), beta.hat.ols[2] + qt(0.9
```

```
## [1] 52.44495 53.25400
```

```
# CI for mean price of that diamond
x <- matrix(c(1, 1.5, 3,2,5,1.5*5, 1.5*2, 1.5*3), 8,1)
c(t(x)%*%beta.hat.ols - qt(0.975, n-p)*sqrt(MSE.ols*t(x)%*%XX.inv%*%x), t(x)%*%beta.hat
```

```
## [1] 119.5424 119.8798
```

```
# PI
c(t(x)%*%beta.hat.ols - qt(0.975, n-p)*sqrt(MSE.ols*(1+t(x)%*%XX.inv%*%x)), t(x)%*%beta.hat.ols +
```

```
## [1] 110.3587 129.0634
```

```
### WLS inferences
XW <- matrix(0,p,n)
tX <- t(X)
for(j in 1:n){
  Wj <- matrix(0,n,1)
  Wj[j] <- W[j]
for(i in 1:p){
  XW[i,j] <- tX[i,]%*%Wj
}
}

XWX.inv <- solve(XW%*%X)



beta.hat.wls <- XWX.inv%*%XW%*%Y
beta.hat.wls
```

```
##                           [,1]
## (Intercept)         3.82491459
## carat              54.59491779
## cut.num            -0.08955524
## color.num           0.15452811
## clarity.num        -0.32964027
## carat:clarity.num   5.21950075
## carat:color.num    -3.11506184
## carat:cut.num       1.23564947
```

```
MSE.wls <- (1/(n-p))*sum(W*(Y - X%*%beta.hat.wls)^2)
MSE.wls
```

```
## [1] 21.00181
```

```
c(beta.hat.wls[2] - qt(0.975, n-p)*sqrt(MSE.wls*XWX.inv[2,2]), beta.hat.wls[2] + qt(0.975, n-p)*s
```

```
##    carat    carat
## 54.20731 54.98253
```

```
c(t(x)%*%beta.hat.wls - qt(0.975, n-p)*sqrt(MSE.wls*t(x)%*%XWX.inv%*%x), t(x)%*%beta.ha
```

```
## [1] 119.2983 119.6437
```

```
c(t(x)%*%beta.hat.wls - qt(0.975, n-p)*sqrt(MSE.wls*(1+t(x)%*%XWX.inv%*%x)), t(x)%*%bet
```

```
## [1] 110.4870 128.4549
```

```
c(t(x)%*%beta.hat.wls - qt(0.975, n-p)*sqrt(MSE.wls*(1.5+t(x)%*%XWX.inv%*%x)), t(x)%*%
```

```
## [1] 108.4686 130.4733
```

14.10 Using built-in functions in R for inference

Make sure to use the weights option in predict.lm for predicting a new response based on the WLS regression model.

```
confint(full.lm.sqrt)
```

```
##                         2.5 %     97.5 %
## (Intercept)         4.9753505  5.7360816
## carat              52.4449462 53.2540030
## cut.num            -0.3098125 -0.1615203
## color.num           0.4042664  0.4988584
## clarity.num        -0.7474246 -0.6521849
## carat:clarity.num   5.5803670  5.6922968
## carat:color.num    -3.4420565 -3.3433480
## carat:cut.num       1.3198461  1.4755679
```

```
confint(wls)
```

```
##                         2.5 %      97.5 %
## (Intercept)         3.5510712  4.09875801
## carat              54.2073087 54.98252686
## cut.num            -0.1424549 -0.03665554
## color.num           0.1211232  0.18793297
## clarity.num        -0.3632933 -0.29598725
## carat:clarity.num   5.1655068  5.27349466
## carat:color.num    -3.1621770 -3.06794668
## carat:cut.num       1.1607589  1.31054002
```

```
new.carat = 1.5
new.cut = 3
new.clarity = 5
new.color = 2
new.data <- data.frame(carat = new.carat, cut.num = new.cut, color.num = new.color, clarity.num =
predict.lm(wls, newdata = new.data, interval = 'confidence')
```

```
##       fit      lwr      upr
## 1 119.471 119.2983 119.6437
```

```
predict.lm(wls, newdata = new.data, interval = 'prediction')
```

```
## Warning in predict.lm(wls, newdata = new.data, interval = "prediction"): Assuming constant pre
```

```
##       fit     lwr      upr
## 1 119.471 110.487 128.4549
```

```
predict.lm(wls, newdata = new.data, interval = 'prediction', weights = 1/1.5)
```

```
##       fit      lwr      upr
## 1 119.471 108.4686 130.4733
```

```
predict.lm(full.lm.sqrt, newdata = new.data, interval = 'confidence')
```

```
##        fit      lwr      upr
## 1 119.7111 119.5424 119.8798
```

```
predict.lm(full.lm.sqrt, newdata = new.data, interval = 'prediction')
```

```
##        fit      lwr      upr
## 1 119.7111 110.3587 129.0634
```

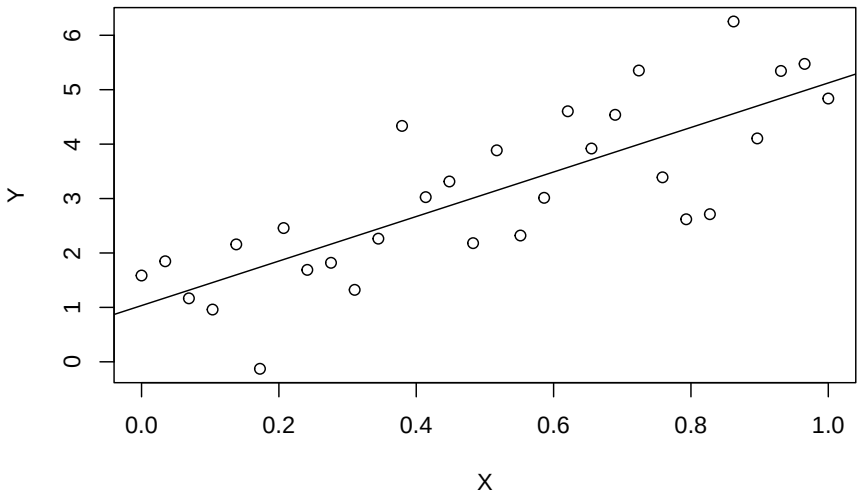
14.11 Leverage, outliers, and influence

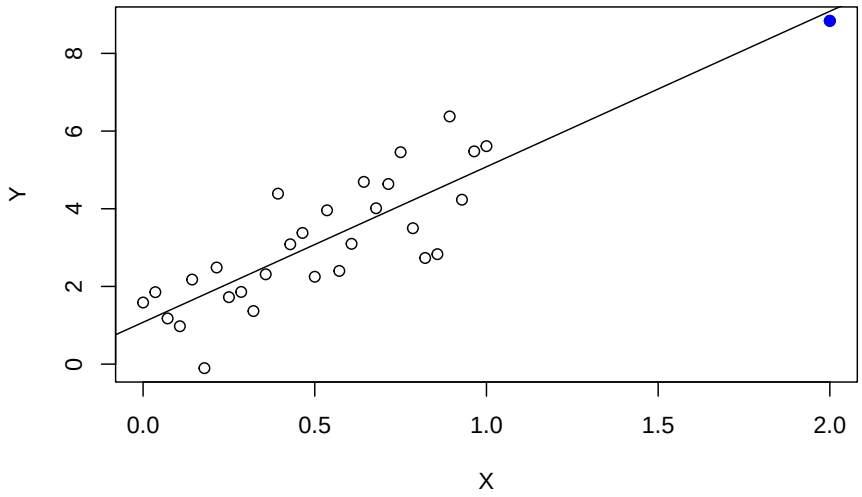
Outlier are observations with very large residuals—the model is not good at predicting the prices of these diamonds. Large residuals may happen by chance, because the model is missing important covariate information, or because one or more assumptions are violated. Often, we consider removing observations with large residuals based on the rationale that if the model is not good at predicting those responses then those observations should not be modeled—a rationale

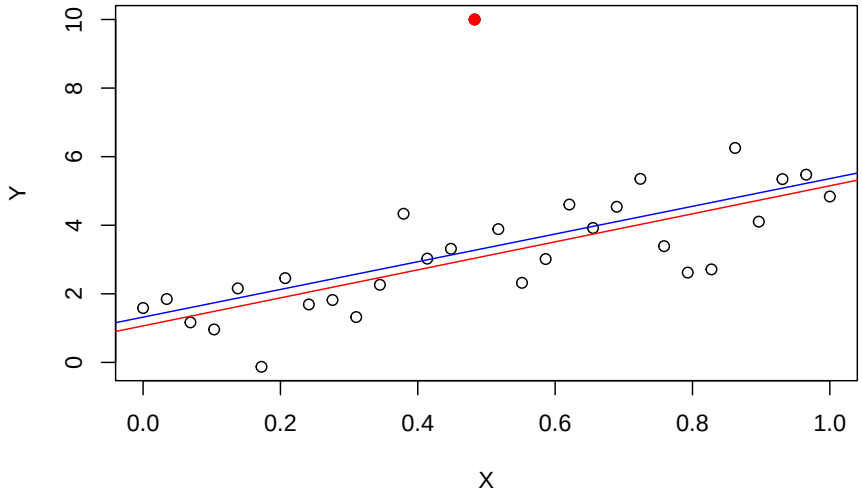
that is, at least partially, flawed. Outlier observations are only troublesome if they have an outsized effect on the fit of the model, which may be measured by comparing $\hat{\beta}$ with and without the observation in question, or comparing $\hat{Y}$ or the SSE. Observations that strongly affect the fit of the model have high *leverage*. Outliers with high leverage are *influential* points. It is only these influential points we need to worry about in terms of whether or not to include them in the model at all. On the other hand, outliers that are not influential are only important when it comes to interpreting the fit of the model at that particular observation, and not for interpreting the model in general.

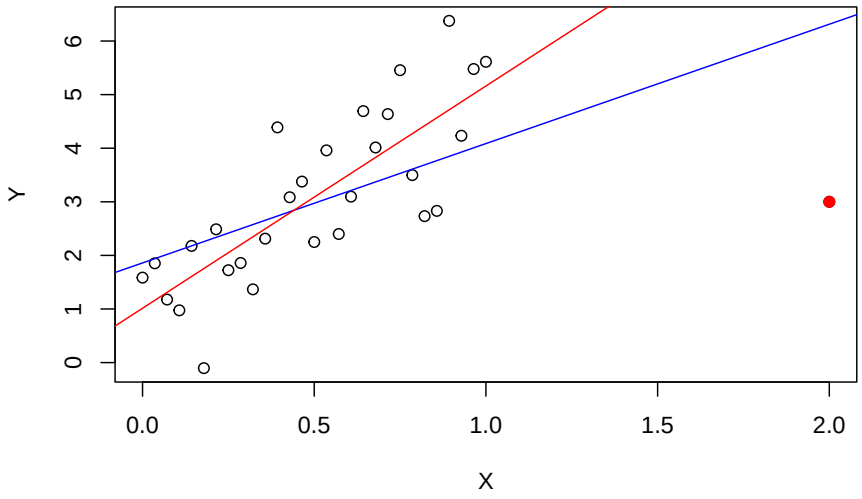
14.11.1 Illustrations

First, some "cartoons" or "toy" examples. The following plots illustrate outliers, leverage, and influence using a simple linear regression on contrived data. The first plot shows a fit without any outliers. The second shows a fit with a high leverage point, that is not an outlier. The third show a pair of fitterd lines with and without a non-influential outlier (one without leverage). And, the fourth shows a plot of a pair of fitted lines with and without an influential point.




Y
0
2
4
6
8
0.0
0.5
1.0
1.5
2.0
X


Y
0
2
4
6
8
10
0.0
0.2
0.4
0.6
0.8
1.0
X



14.12 Numerical summaries of outliers, leverage, and influence

The "hat" matrix $H = X(X^\top X)^{-1}X^\top$ determines high leverage points. To see this, note that the fitted responses are given by $\hat{Y} = HY$, which are linear combinations of all the responses. We can express $\hat{Y}_i$—the predicted price of the i^{th} diamond—as a linear combination of the price of the i^{th} diamond itself and the other $n-1$ diamonds: $\hat{Y}_i = h_{ii}Y_i + \sum_{i \neq j} h_{ij}Y_j$ where h_{ij} is the ij entry of H. If we simply define leverage to be the degree to which an observed response determines its own prediction (or fitted value) then h_{ii}—the diagonal of H—defines leverage.

```
h.X <- hat(X)
h.X
```

```
##  [1] 0.09682659 0.08884843 0.08140573 0.07449847 0.06812665 0.06229028
##  [7] 0.05698936 0.05222389 0.04799386 0.04429928 0.04114014 0.03851646
## [13] 0.03642821 0.03487542 0.03385807 0.03337617 0.03342971 0.03401870
## [19] 0.03514314 0.03680303 0.03899836 0.04172914 0.04499536 0.04879703
## [25] 0.05313415 0.05800671 0.06341472 0.06935818 0.07583708 0.47463768
```

```
# hat(model.matrix(Y~X))   # same
which(h.X > (3*2 / 30))   # 3(k+1)/n
```

```
## [1] 30
```

```
which.max(h.X)
```

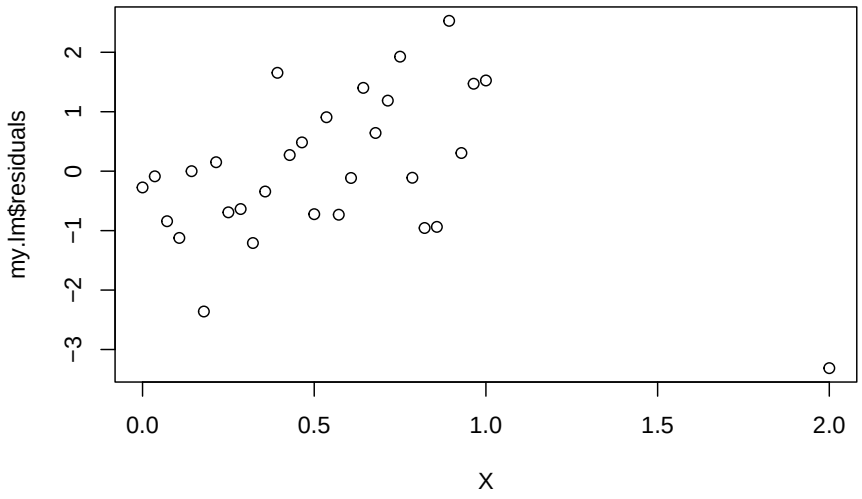
```
## [1] 30
```

```
# "by hand"
X.d <- cbind(rep(1,30),X)
hat.X <- X.d%*%solve(t(X.d)%*%X.d)%*%t(X.d)  # n by n matrix
diag(hat.X)
```

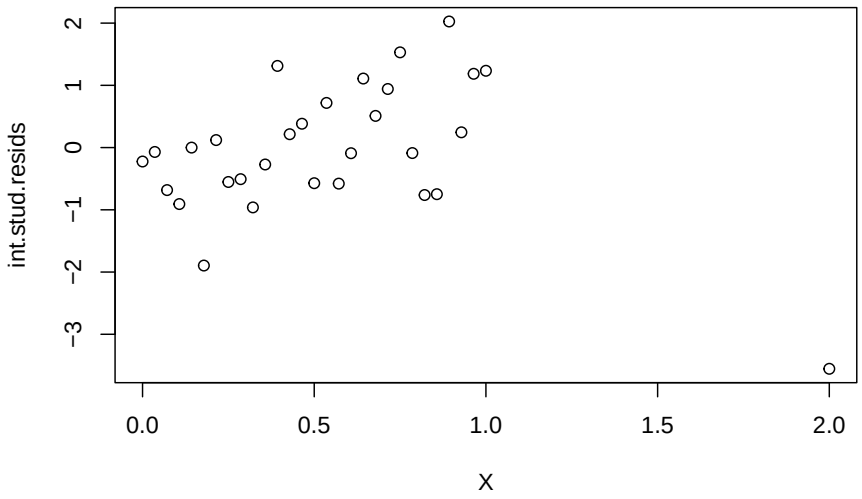
```
##  [1] 0.09682659 0.08884843 0.08140573 0.07449847 0.06812665 0.06229028
##  [7] 0.05698936 0.05222389 0.04799386 0.04429928 0.04114014 0.03851646
## [13] 0.03642821 0.03487542 0.03385807 0.03337617 0.03342971 0.03401870
## [19] 0.03514314 0.03680303 0.03899836 0.04172914 0.04499536 0.04879703
## [25] 0.05313415 0.05800671 0.06341472 0.06935818 0.07583708 0.47463768
```

Outliers are defined by large residuals. In multiple linear regression there is more than one way to define residuals and studentized residuals. "Studentized" refers to normalizing residuals by the estimated standard deviation. Internally studentized residuals include each observation in the calculation of the estimated standard deviation whereas externally studentized residuals ignore the corresponding observation when computing the estimated standard deviation for normalization.

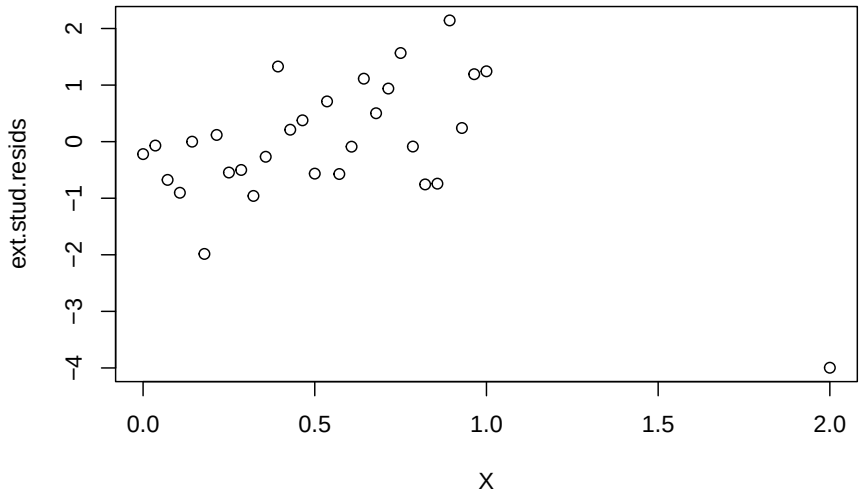
```
plot(X, my.lm$residuals)
```




```
MSE <- sum(my.lm$residuals^2)/(30-2)
int.stud.resids <- my.lm$residuals / sqrt(MSE * (1-h.X))
plot(X, int.stud.resids)
```



```
MSE.i <- rep(NA, 30)
for(i in 1:30){
MSE.i[i] <- sum(my.lm$residuals[-i]^2)/(30-2-1)
}
ext.stud.resids <- my.lm$residuals / sqrt(MSE.i * (1-h.X))
plot(X, ext.stud.resids)
```



An observation's *influence* can be defined by the degree to which is affects the model through $\hat{\beta}$ or $\hat{Y}$. Cook's D and DfFits measure the fit of the model, which is related to residuals and fitted/predicted values, whereas DfBetas measures the change in the fitted coefficients with or without the given observation.

```
# base r functions
cd <- cooks.distance(my.lm)
which(cd > 2*sqrt(2/30))
```

```
## 30
## 30
```

```
dfb <- dfbetas(my.lm)
which(dfb > 2)
```

```
## [1] 30
```

```
dff <- dffits(my.lm)
which(dff > 2*sqrt(2/30))
```

```
## 26
## 26
```

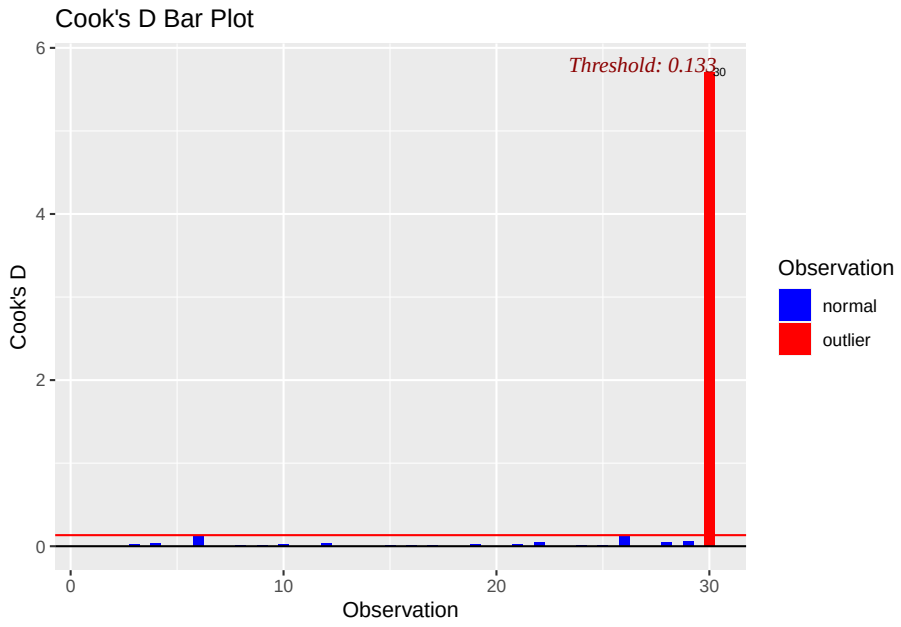
```
library(olsrr)
```

```
## Warning: package 'olsrr' was built under R version 4.2.2
```

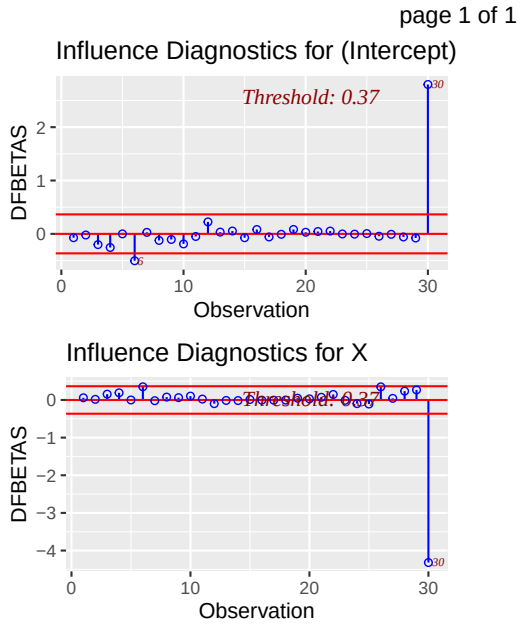
```
##
## Attaching package: 'olsrr'
```

```
## The following object is masked from 'package:datasets':
##
##     rivers
```

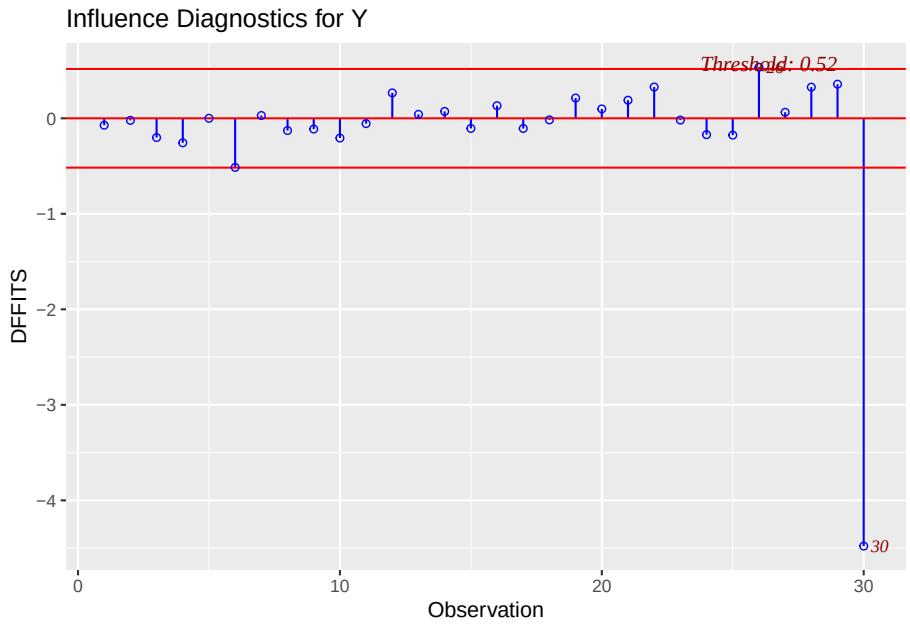
```
ols_plot_cooksd_bar(my.lm)  # lol
```




```
ols_plot_dfbetas(my.lm)
```




```
ols_plot_dffits(my.lm)
```



14.13 Leverage, Diamonds model

```
wls <- lm(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num + carat*c

MSE.wls <- sum((wls$residuals)^2)/(nrow(diamonds)-8)

leverages.lm <- lm.influence(wls)$hat

# k + 1  = 8 fitted regression coefficients
# n = 53940

3 * 8/53940
```

```
## [1] 0.0004449388
```

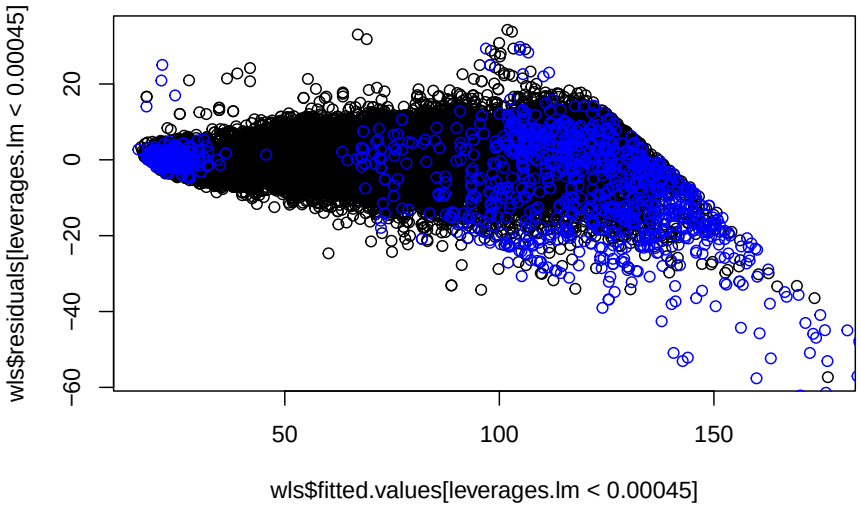
```
sum(leverages.lm > 0.00045)
```

```
## [1] 1248
```

```
sum(leverages.lm > 0.00045) / 53940
```

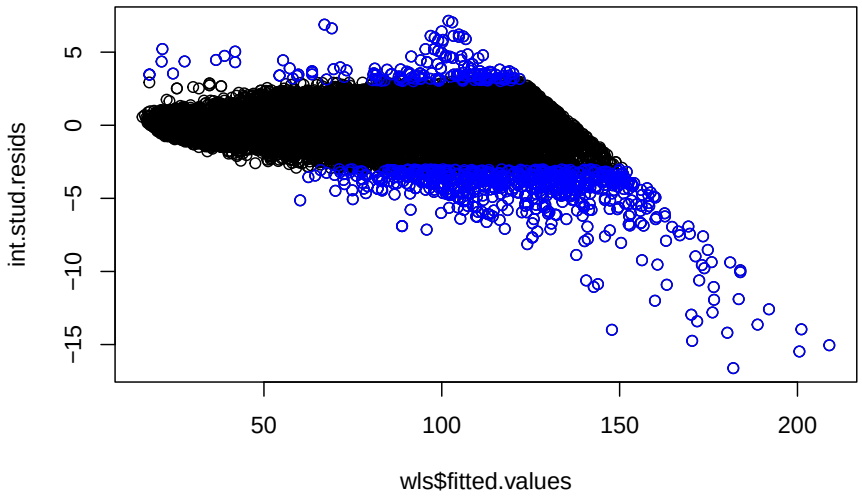
```
## [1] 0.02313682
```

```
plot(wls$fitted.values[leverages.lm < 0.00045], wls$residuals[leverages.lm < 0.00045])
points(wls$fitted.values[leverages.lm > 0.00045], wls$residuals[leverages.lm > 0.00045]
```

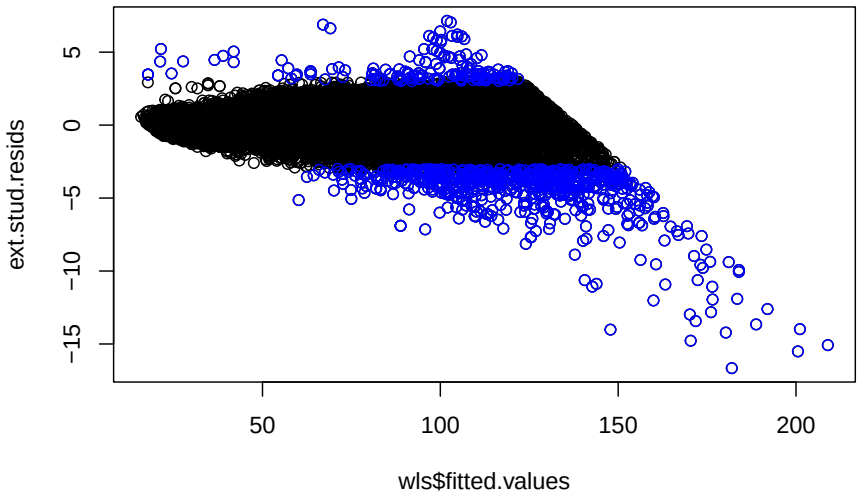


14.13.1 Outliers, diamonds model

```
int.stud.resids <- wls$residuals / sqrt(MSE.wls * (1-leverages.lm))
plot(wls$fitted.values, int.stud.resids)
points(wls$fitted.values[abs(int.stud.resids)>3], int.stud.resids[abs(int.stud.resids)>3], col =
```

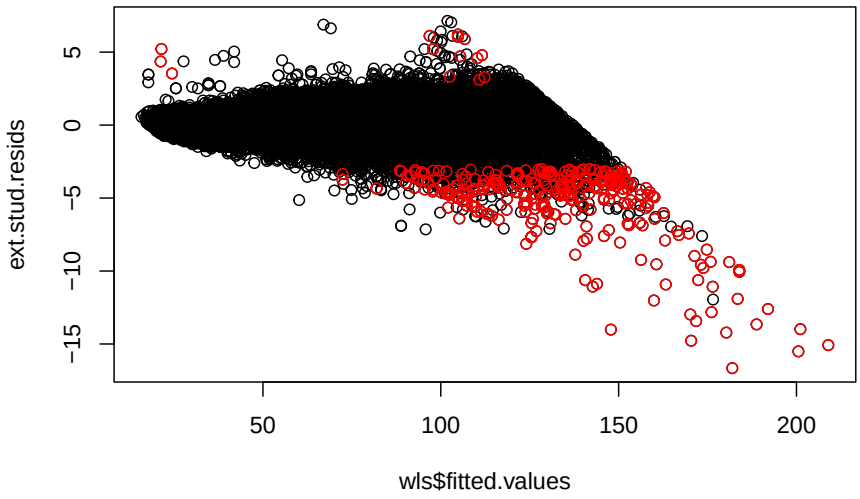



```
MSE.i <- rep(NA, 53940)
for(i in 1:53940){
MSE.i[i] <- sum(wls$residuals[-i]^2)/(53940-8-1)
}
ext.stud.resids <- wls$residuals / sqrt(MSE.i * (1-leverages.lm))
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[abs(ext.stud.resids)>3], ext.stud.resids[abs(ext.stud.resids)
```



14.13.2 plotting outliers with leverage

```
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[abs(ext.stud.resids)>3 & leverages.lm > 0.00045], ext.stud.resids[abs(ex
```




```
length(wls$fitted.values[abs(ext.stud.resids)>3 & leverages.lm > 0.00045])
```

```
## [1] 308
```

```
length(wls$fitted.values[abs(ext.stud.resids)>3 & leverages.lm > 0.00045])/53940
```

```
## [1] 0.005710048
```

14.13.3 Cook's distance, DF betas, DF fits

```
# base r functions
cd <- cooks.distance(wls, weights  = 1/diamonds$carat)
sum(cd > 2*sqrt(2/53940))
```

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

```
dfb <- dfbetas(wls, weights  = 1/diamonds$carat)
sum(abs(dfb) > 2*sqrt(2/53940))
```

```
## [1] 13647
```

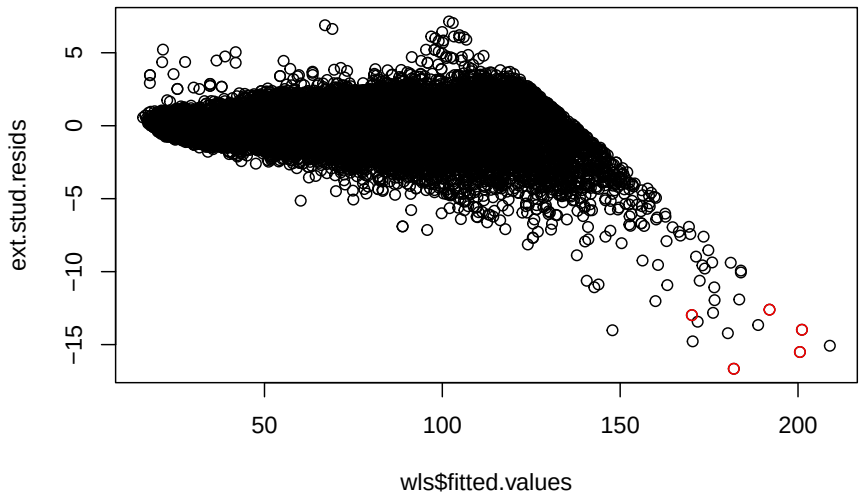
```
dff <- dffits(wls)
sum(abs(dff) > 2*sqrt(2/53940))
```

```
## [1] 12754
```

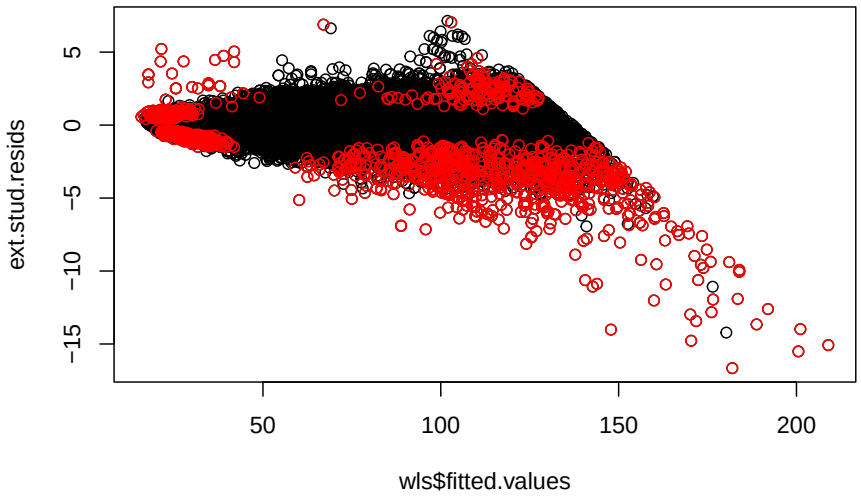
```
my.dff <- ext.stud.resids*sqrt(leverages.lm/(1-leverages.lm))
sum(my.dff > 2*sqrt(2/53940))
```

```
## [1] 4849
```

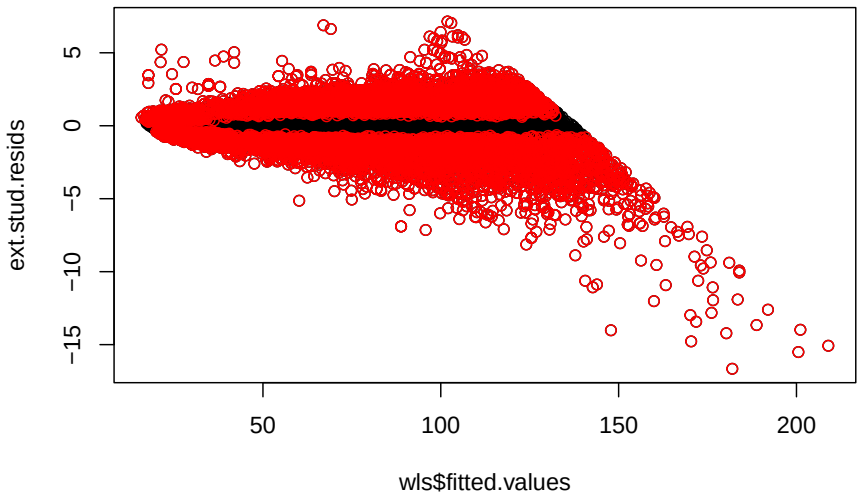
```
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[cd > 2*sqrt(2/53940)], ext.stud.resids[cd > 2*sqrt(2/53940)], col = 'red
```



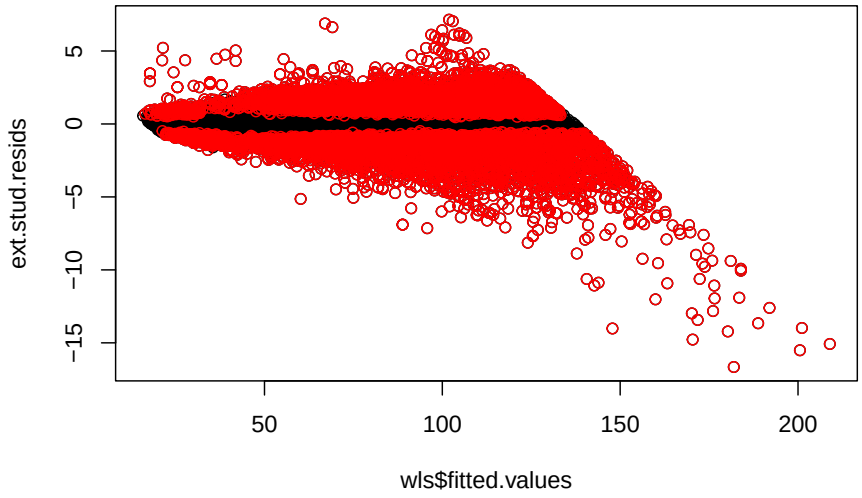

```
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[abs(dfb) > 2*sqrt(2/53940)], ext.stud.resids[abs(dfb) > 2*sqrt(2/53940)]
```




```
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[abs(dff) > 2*sqrt(2/53940)], ext.stud.resids[abs(dff) > 2*sqrt
```



```
plot(wls$fitted.values, ext.stud.resids)
points(wls$fitted.values[abs(my.dff) > 2*sqrt(2/53940)], ext.stud.resids[abs(my.dff) > 2*sqrt(2/5
```



14.13.4 Model fit without high leverage outliers

```
summary(wls)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut.num + color.num + clarity.num +
##     carat * clarity.num + carat * color.num + carat * cut.num,
##     data = diamonds, weights = 1/diamonds$carat)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -45.644  -2.700  -0.031   2.874  42.921
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)        3.82491    0.13972   27.376  < 2e-16 ***
## carat             54.59492    0.19776  276.068  < 2e-16 ***
```

```
## cut.num           -0.08956    0.02699   -3.318 0.000907 ***
## color.num          0.15453    0.01704    9.067  < 2e-16 ***
## clarity.num       -0.32964    0.01717  -19.199  < 2e-16 ***
## carat:clarity.num  5.21950    0.02755  189.471  < 2e-16 ***
## carat:color.num   -3.11506    0.02404 -129.588  < 2e-16 ***
## carat:cut.num      1.23565    0.03821   32.339  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.583 on 53932 degrees of freedom
## Multiple R-squared:  0.9764, Adjusted R-squared:  0.9764
## F-statistic: 3.192e+05 on 7 and 53932 DF,  p-value: < 2.2e-16
```

```
diamonds2 <- diamonds[!(abs(ext.stud.resids)>3 & leverages.lm > 0.00045),]
wls2 <- lm(sqrt(price)~carat+cut.num+color.num+clarity.num + carat*clarity.num + carat

summary(wls2)
```

```
##
## Call:
## lm(formula = sqrt(price) ~ carat + cut.num + color.num + clarity.num +
##     carat * clarity.num + carat * color.num + carat * cut.num,
##     data = diamonds2, weights = 1/diamonds2$carat)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -33.925  -2.683  -0.059   2.760  38.284
##
## Coefficients:
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)        2.96547    0.13626   21.764  < 2e-16 ***
## carat             55.93431    0.19645  284.727  < 2e-16 ***
## cut.num            0.03250    0.02617    1.242    0.214
## color.num          0.09730    0.01644    5.920 3.25e-09 ***
## clarity.num       -0.24500    0.01671  -14.666  < 2e-16 ***
## carat:clarity.num  5.09844    0.02727  186.976  < 2e-16 ***
## carat:color.num   -3.02464    0.02343 -129.071  < 2e-16 ***
## carat:cut.num      1.03596    0.03754   27.596  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.365 on 53624 degrees of freedom
## Multiple R-squared:  0.9784, Adjusted R-squared:  0.9784
## F-statistic: 3.475e+05 on 7 and 53624 DF,  p-value: < 2.2e-16
```

```
confint(wls)
```

```
##                         2.5 %      97.5 %
## (Intercept)         3.5510712  4.09875801
## carat              54.2073087 54.98252686
## cut.num            -0.1424549 -0.03665554
## color.num           0.1211232  0.18793297
## clarity.num        -0.3632933 -0.29598725
## carat:clarity.num   5.1655068  5.27349466
## carat:color.num    -3.1621770 -3.06794668
## carat:cut.num       1.1607589  1.31054002
```

```
new.carat = 1.5
new.cut = 3
new.clarity = 5
new.color = 2
new.data <- data.frame(carat = new.carat, cut.num = new.cut, color.num = new.color, clarity.num =
predict.lm(wls, newdata = new.data, interval = 'confidence')
```

```
##       fit      lwr      upr
## 1 119.471 119.2983 119.6437
```

```
predict.lm(wls, newdata = new.data, interval = 'prediction', weights = 1.5)
```

```
##       fit      lwr     upr
## 1 119.471 112.1349 126.807
```

```
confint(wls2)
```

```
##                         2.5 %      97.5 %
## (Intercept)         2.69840297  3.23252928
## carat              55.54926937 56.31935301
## cut.num            -0.01880028  0.08380572
## color.num           0.06508121  0.12951112
## clarity.num        -0.27774342 -0.21225746
## carat:clarity.num   5.04499842  5.15188930
## carat:color.num    -3.07056660 -2.97870518
## carat:cut.num       0.96238548  1.10954286
```

```
new.carat = 1.5
new.cut = 3
new.clarity = 5
```

```
new.color = 2
new.data <- data.frame(carat = new.carat, cut.num = new.cut, color.num = new.color, cla
predict.lm(wls2, newdata = new.data, interval = 'confidence')
```

```
##        fit      lwr      upr
## 1 119.7603 119.5934 119.9272
```

```
predict.lm(wls2, newdata = new.data, interval = 'prediction', weights = 1.5)
```

```
##        fit      lwr      upr
## 1 119.7603 112.7735 126.7471
```

14.14 Dealing with highly influential data points

1. Do not simply delete and ignore influential data points.
2. Perform analyses both with and without the subset of points you identify as unusually influential.
3. Recommend further analysis be done to understand why some points are not well-explained by the model.
    a. Are there data-entry mistakes?
    b. Are there important explanatory variables missing from the model?
    c. Are these all "exceptional" cases?

Question: Suppose this model is being used as an automatic diamond pricing algorithm by a diamond seller. A new diamond is "fed" into the model, and out comes a suggested price. What should be the impact of the above influence analysis on this pricing algorithm?