Statistical Methods II

Derek L. Sonderegger

November 19, 2020

Contents

Appendix   257

Preface

These notes are intended to be used in the second semester of a two-semester sequence of Statistical Methodology. We assume that students have seen t-tests, Simple Regression, and ANOVA. The second semester emphasizes the uniform matrix notation $y = X\beta + \epsilon$ and the interpretation of the coefficients. We cover model diagnostics, transformation, model selection, interactions of continuous and categorical predictors as well as introduce random effects in the experimental design context. Finally we introduce logistic regression.

Statistical Theory

Chapter 1

Matrix Manipulation

Learning Outcomes

- Perform basic vector and matrix operations of addition and multiplication
- Perform matrix operations of transpose and inverse.

1.1 Introduction

Almost all of the calculations done in classical statistics require formulas with large number of subscripts and many different sums. In this chapter we will develop the mathematical machinery to write these formulas in a simple compact formula using *matrices*.

1.2 Types of Matrices

We will first introduce the idea behind a matrix and give several special types of matrices that we will encounter.

1.2.1 Scalars

To begin, we first define a *scalar*. A scalar is just a single number, either real or complex. The key is that a scalar is just a single number. For example, 6 is a scalar, as is -3. By convention, variable names for scalars will be lower case and not in bold typeface.

Examples could be $a = 5$, $b = \sqrt{3}$, or $\sigma = 2$.

1.2.2 Vectors

A vector is collection of scalars, arranged as a row or column. Our convention will be that a vector will be a lower cased letter but written in a bold type. In other branches of mathematics is common to put a bar over the variable name to denote that it is a vector, but in statistics, we have already used a bar to denote a mean.

Examples of column vectors could be

$$a = \begin{bmatrix} 2 \\ -3 \\ 4 \end{bmatrix} \qquad b = \begin{bmatrix} 2 \\ 8 \\ 3 \\ 4 \\ 1 \end{bmatrix}$$

and examples of row vectors are

$$c = \begin{bmatrix} 8 & 10 & 43 & -22 \end{bmatrix}$$

$$d = \begin{bmatrix} -1 & 5 & 2 \end{bmatrix}$$

To denote a specific entry in the vector, we will use a subscript. For example, the second element of d is $d_2 = 5$. Notice, that we do not bold this symbol because the second element of the vector is the scalar value 5.

1.2.3 Matrix

Just as a vector is a collection of scalars, a matrix can be viewed as a collection of vectors (all of the same length). We will denote matrices with bold capitalized letters. In general, I try to use letters at the end of the alphabet for matrices. Likewise, I try to use symmetric letters to denote symmetric matrices.

For example, the following is a matrix with two rows and three columns

$$W = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

and there is no requirement that the number of rows be equal, less than, or greater than the number of columns. In denoting the size of the matrix, we first refer to the number of rows and then the number of columns. Thus W is a 2×3 matrix and it sometimes is helpful to remind ourselves of this by writing $W_{2\times3}$.

To pick out a particular element of a matrix, I will again use a subscripting notation, always with the row number first and then column. Notice the notational shift to lowercase, non-bold font.

$$w_{1,2} = 2 \qquad \text{and} \qquad w_{2,3} = 6$$

There are times I will wish to refer to a particular row or column of a matrix and we will use the following notation

$$w_{1,\cdot} = \left[\begin{array}{ccc} 1 & 2 & 3 \end{array} \right]$$

is the first row of the matrix W. The second column of matrix W is

$$w_{\cdot,2} = \left[\begin{array}{c} 2 \\ 5 \end{array} \right]$$

1.2.4 Square Matrices

A square matrix is a matrix with the same number of rows as columns. The following are square

$$Z = \left[\begin{array}{cc} 3 & 6 \\ 8 & 10 \end{array} \right] \qquad X = \left[\begin{array}{ccc} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{array} \right]$$

1.2.5 Symmetric Matrices

In statistics we are often interested in square matrices where the i, j element is the same as the j, i element. For example, $x_{1,2} = x_{2,1}$ in the above matrix X.

Consider a matrix D that contains the distance from four towns to each of the other four towns. Let $d_{i,j}$ be the distance from town i to town j. It only makes sense that the distance doesn't matter which direction you are traveling, and we should therefore require that $d_{i,j} = d_{j,i}$.

In this example, it is the values $d_{i,i}$ represent the distance from a town to itself, which should be zero. It turns out that we are often interested in the terms $d_{i,i}$ and I will refer to those terms as the *main diagonal* of matrix D.

Symmetric matrices play a large role in statistics because matrices that represent the covariances between random variables must be symmetric because $Cov\,(Y, Z) = Cov\,(Z, Y)$.

1.2.6 Diagonal Matrices

A square matrix that has zero entries in every location except the main diagonal is called a diagonal matrix. Here are two examples:

$$Q = \left[\begin{array}{ccc} 4 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 6 \end{array} \right] \qquad R = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{array} \right]$$

Sometimes to make matrix more clear, I will replace the 0 with a dot to empha-size the non-zero components.

$$R = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 2 & \cdot & \cdot \\ \cdot & \cdot & 2 & \cdot \\ \cdot & \cdot & \cdot & 3 \end{bmatrix}$$

1.2.7   Identity Matrices

A diagonal matrix with main diagonal values exactly 1 is called the identity matrix. The 3×3 identity matrix is denoted I_3.

$$I_3 = \begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \\ \cdot & \cdot & 1 \end{bmatrix}$$

1.3   Operations on Matrices

1.3.1   Transpose

The simplest operation on a square matrix matrix is called *transpose*. It is defined as $M = W^T$ if and only if $m_{i,j} = w_{j,i}$.

$$Z = \begin{bmatrix} 1 & 6 \\ 8 & 3 \end{bmatrix} \qquad Z^T = \begin{bmatrix} 1 & 8 \\ 6 & 3 \end{bmatrix}$$

$$M = \begin{bmatrix} 3 & 1 & 2 \\ 9 & 4 & 5 \\ 8 & 7 & 6 \end{bmatrix} \qquad M^T = \begin{bmatrix} 3 & 9 & 8 \\ 1 & 4 & 7 \\ 2 & 5 & 6 \end{bmatrix}$$

We can think of this as swapping all elements about the main diagonal. Alter-natively we could think about the transpose as making the first row become the first column, the second row become the second column, etc. In this fashion we could define the transpose of a non-square matrix.

$$W = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$W^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

1.3.2 Addition and Subtraction

Addition and subtraction are performed *element-wise*. This means that two matrices or vectors can only be added or subtracted if their dimensions match.

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} + \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 6 \\ 8 \\ 10 \\ 12 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 8 \\ 2 & 4 \\ 11 & 15 \end{bmatrix} - \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & -6 \end{bmatrix} = \begin{bmatrix} 4 & 6 \\ -1 & 0 \\ 6 & 21 \end{bmatrix}$$

1.3.3 Multiplication

Multiplication is the operation that is vastly different for matrices and vectors than it is for scalars. There is a great deal of mathematical theory that suggests a useful way to define multiplication. What is presented below is referred to as the *dot-product* of vectors in calculus, and is referred to as the standard *inner-product* in linear algebra.

1.3.4 Vector Multiplication

We first define multiplication for a row and column vector. For this multiplication to be defined, both vectors must be the same length. The product is the sum of the element-wise multiplications.

$$\begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \end{bmatrix} = (1 \cdot 5) + (2 \cdot 6) + (3 \cdot 7) + (4 \cdot 8) = 5 + 12 + 21 + 32 = 70$$

1.3.5 Matrix Multiplication

Matrix multiplication is just a sequence of vector multiplications. If X is a $m \times n$ matrix and W is $n \times p$ matrix then $Z = XW$ is a $m \times p$ matrix where $z_{i,j} = x_{i,\cdot} w_{\cdot,j}$ where $x_{i,\cdot}$ is the ith column of X and $w_{\cdot,j}$ is the jth column of W. For example, let

$$X = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix} \qquad W = \begin{bmatrix} 13 & 14 \\ 15 & 16 \\ 17 & 18 \\ 19 & 20 \end{bmatrix}$$

so X is $3{\times}4$ (which we remind ourselves by adding a $3{\times}4$ subscript to X as $X_{3\times4}$) and W is $W_{4\times2}$. Because the *inner* dimensions match for this multiplication, then $Z_{3\times2} = X_{3\times4}W_{4\times2}$ is defined where

$$z_{1,1} = x_{1,\cdot}w_{\cdot,1}$$
$$= (1 \cdot 13) + (2 \cdot 15) + (3 \cdot 17) + (4 \cdot 19) = 170$$

and similarly

$$z_{2,1} = x_{2,\cdot}w_{\cdot,1}$$
$$= (5 \cdot 13) + (6 \cdot 15) + (7 \cdot 17) + (8 \cdot 19)$$
$$= 426$$

so that

$$Z = \left[\begin{array}{cc} 170 & 180 \\ 426 & 452 \\ 682 & 724 \end{array}\right]$$

For another example, we note that

$$\left[\begin{array}{ccc} 1 & 2 & 3 \\ 2 & 3 & 4 \end{array}\right]\left[\begin{array}{cc} 1 & 2 \\ 2 & 2 \\ 1 & 2 \end{array}\right] = \left[\begin{array}{cc} 1+4+3 & 2+4+6 \\ 2+6+4 & 4+6+8 \end{array}\right] = \left[\begin{array}{cc} 8 & 12 \\ 12 & 18 \end{array}\right]$$

Notice that this definition of multiplication means that the order matters. Above, we calculated $X_{3\times4}W_{4\times2}$ but we cannot reverse the order because the inner dimensions do not match up.

1.3.6   Scalar times a Matrix

Strictly speaking, we are not allowed to multiply a matrix by a scalar because the dimensions do not match. However, it is often notationally convenient. So we define aX to be the *element-wise* multiplication of each element of X by the scalar a. Because this is just a notational convenience, the mathematical theory about inner-products does not apply to this operation.

$$5\left[\begin{array}{cc} 4 & 5 \\ 7 & 6 \\ 9 & 10 \end{array}\right] = \left[\begin{array}{cc} 20 & 25 \\ 35 & 30 \\ 45 & 50 \end{array}\right]$$

Because of this definition, it is clear that $aX = Xa$ and the order does not matter. Thus when mixing scalar multiplication with matrices, it is acceptable to reorder scalars, but not matrices.

1.3.7 Determinant

The determinant is defined only for square matrices and can be thought of as the matrix equivalent of the absolute value or magnitude (i.e. $|-6| = 6$). The determinant gives a measure of the multi-dimensional size of a matrix (say the matrix A) and as such is denoted $\det(A)$ or $|A|$. Generally this is a very tedious thing to calculate by hand and for completeness sake, we will give a definition and small examples.

For a 2×2 matrix

$$\begin{vmatrix} a & c \\ b & d \end{vmatrix} = ad - cb$$

So a simple example of a determinant is

$$\begin{vmatrix} 5 & 2 \\ 3 & 10 \end{vmatrix} = 50 - 6 = 44$$

The determinant can be thought of as the area of the parallelogram created by the row or column vectors of the matrix.

1.3.8 Inverse

In regular algebra, we are often interested in solving equations such as

$$5x = 15$$

for x. To do so, we multiply each side of the equation by the inverse of 5, which is $1/5$.

$$5x = 15$$
$$\frac{1}{5} \cdot 5 \cdot x = \frac{1}{5} \cdot 15$$
$$1 \cdot x = 3$$
$$x = 3$$

For scalars, we know that the inverse of scalar a is the value that when multiplied by a is 1. That is we see to find a^{-1} such that $aa^{-1} = 1$.

In the matrix case, I am interested in finding A^{-1} such that $A^{-1}A = I$ and $AA^{-1} = I$. For both of these multiplications to be defined, A must be a square matrix and so the inverse is only defined for square matrices.

For a 2×2 matrix

$$W = \left[\begin{array}{cc} a & b \\ c & d \end{array} \right]$$

the inverse is given by:

$$W^{-1} = \frac{1}{\det W} \left[\begin{array}{cc} d & -b \\ -c & a \end{array} \right]$$

For example, suppose

$$W = \left[\begin{array}{cc} 1 & 2 \\ 5 & 3 \end{array} \right]$$

then $\det W = 3 - 10 = -7$ and

$$W^{-1} = \frac{1}{-7} \left[\begin{array}{cc} 3 & -2 \\ -5 & 1 \end{array} \right]$$

$$= \left[\begin{array}{cc} -\frac{3}{7} & \frac{2}{7} \\ \frac{5}{7} & -\frac{1}{7} \end{array} \right]$$

and thus

$$WW^{-1} = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} -\frac{3}{7} & \frac{2}{7} \\ \frac{5}{7} & -\frac{1}{7} \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{3}{7} + \frac{10}{7} & \frac{2}{7} - \frac{2}{7} \\ -\frac{15}{7} + \frac{15}{7} & \frac{10}{7} - \frac{3}{7} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I_2$$

Not every square matrix has an inverse. If the determinant of the matrix (which we think of as some measure of the magnitude or *size* of the matrix) is zero, then the formula would require us to divide by zero. Just as we cannot find the inverse of zero (i.e. solve $0x = 1$ for x), a matrix with zero determinate is said to have no inverse.

1.4   Exercises

Consider the following matrices:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 5 & 4 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 6 & 4 & 3 \\ 8 & 7 & 6 \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \qquad \mathbf{d} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \qquad \mathbf{E} = \begin{bmatrix} 1 & 2 \\ 2 & 6 \end{bmatrix}$$

1. Find $\mathbf{Bc}$
2. Find $\mathbf{AB}^T$
3. Find $\mathbf{c}^T\mathbf{d}$
4. Find $\mathbf{cd}^T$
5. Confirm that

$$\mathbf{E}^{-1} = \begin{bmatrix} 3 & -1 \\ -1 & 1/2 \end{bmatrix}$$

is the inverse of $\mathbf{E}$ by calculating $\mathbf{EE}^{-1} = \mathbf{I}$.

Chapter 2

Parameter Estimation

```r
library(tidyverse)   # dplyr, tidyr, ggplot2
```

Learning Outcomes

- Write simple regression or one-way ANOVA models as $y \sim N(X\beta, \sigma^2 \mathbf{I}_n)$

- Utilizing R and a sample of data, calculate the estimators

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$$

and

$$\hat{\sigma}^2 = \text{MSE} = \frac{1}{n-p} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Calculate the uncertainty of $\hat{\beta}_j$ as

$$\text{StdErr}\left(\hat{\beta}_j\right) = \sqrt{\hat{\sigma}^2 \left[\left(X^TX\right)^{-1}\right]_{jj}}$$

2.1 Introduction

We have previously looked at ANOVA and regression models and, in many ways, they felt very similar. In this chapter we will introduce the theory that allows us to understand both models as a particular flavor of a larger class of models known as *linear models*.

First we clarify what a linear model is. A linear model is a model where the data (which we will denote using roman letters as x and y) and parameters of interest (which we denote using Greek letters such as α and β) interact only via addition and multiplication. The following are linear models:

Model	Formula
ANOVA	$y_{ij} = \mu + \tau_i + \epsilon_{ij}$
Simple Regression	$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$
Quadratic Term	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i$
General Regression	$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_p x_{i,p} + \epsilon_i$

Notice in the Quadratic model, the square is not a parameter and we can consider x_i^2 as just another column of data. This leads to the second example of multiple regression where we just add more slopes for other covariates where the pth covariate is denoted $x_{\cdot,p}$ and might be some transformation (such as x^2 or $\log x$) of another column of data. The critical point is that the transformation to the data x does not depend on a parameter. Thus the following is *not* a linear model

$$y_i = \beta_0 + \beta_1 x_i^\alpha + \epsilon_i$$

2.2   Model Specifications

2.2.1   Simple Regression

We would like to represent all linear models in a similar compact matrix representation. This will allow us to make the transition between simple and multiple regression (and ANCOVA) painlessly.

To begin, lets consider the simple regression model.

Typically we'll write the model as if we are specifying the ith element of the data set

$$y_i = \underbrace{\beta_0 + \beta_1 x_i}_{\text{signal}} + \underbrace{\epsilon_i}_{\text{noise}} \qquad \text{where } \epsilon_i \stackrel{iid}{\sim} N(0, \sigma^2)$$

Notice we have a data generation model where there is some relationship between the explanatory variable and the response which we can refer to as the "signal" part of the defined model and the noise term which represents unknown actions effecting each data point that move the response variable. We don't know what those unknown or unmeasured effects are, but we do know the sum of those effects results in a vertical shift from the signal part of the model.

Simple Regression Model

This representation of the model implicitly assumes that our data set has n observations and we could write the model using *all* the obsesrvations using matrices and vectors that correspond the the data and the parameters.

$$y_1 = \beta_0 + \beta_1 x_1 + \epsilon_1$$
$$y_2 = \beta_0 + \beta_1 x_2 + \epsilon_2$$
$$y_3 = \beta_0 + \beta_1 x_3 + \epsilon_3$$
$$\vdots$$
$$y_{n-1} = \beta_0 + \beta_1 x_{n-1} + \epsilon_{n-1}$$
$$y_n = \beta_0 + \beta_1 x_n + \epsilon_n$$

where, as usual, $\epsilon_i \overset{iid}{\sim} N\left(0, \sigma^2\right)$. These equations can be written using matrices as

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ \vdots & \vdots \\ 1 & x_{n-1} \\ 1 & x_n \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}}_{\beta} + \underbrace{\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_{n-1} \\ \epsilon_n \end{bmatrix}}_{\epsilon}$$

and we compactly write the model as

$$y = X\beta + \epsilon \quad \text{where} \quad \epsilon \sim N(0, \sigma^2 I_n)$$

where X is referred to as the *design matrix* and β is the vector of *location parameters* we are interested in estimating. To be very general, a vector of random variables needs to describe how each of them varies in relation to each other, so we need to specify the variance *matrix*. However because ϵ_i is independent of

ϵ_j, for all (i, j) pairs, the variance matrix can be written as $\sigma^2 I$ because all of the covariances are zero.

2.2.2   ANOVA model

The anova model is also a linear model and all we must do is create a appropriate design matrix. Given the design matrix X, all the calculations are identical as in the simple regression case.

2.2.2.1   Cell means representation

Recall the cell means representation is

$$y_{i,j} = \mu_i + \epsilon_{i,j}$$

where $y_{i,j}$ is the jth observation within the ith group. To clearly show the creation of the X matrix, let the number of groups be $p = 3$ and the number of observations per group be $n_i = 4$. We now expand the formula to show all the data.

$$y_{1,1} = \mu_1 + \epsilon_{1,1}$$
$$y_{1,2} = \mu_1 + \epsilon_{1,2}$$
$$y_{1,3} = \mu_1 + \epsilon_{1,3}$$
$$y_{1,4} = \mu_1 + \epsilon_{1,4}$$
$$y_{2,1} = \mu_2 + \epsilon_{2,1}$$
$$y_{2,2} = \mu_2 + \epsilon_{2,2}$$
$$y_{2,3} = \mu_2 + \epsilon_{2,3}$$
$$y_{2,4} = \mu_2 + \epsilon_{2,4}$$
$$y_{3,1} = \mu_3 + \epsilon_{3,1}$$
$$y_{3,2} = \mu_3 + \epsilon_{3,2}$$
$$y_{3,3} = \mu_3 + \epsilon_{3,3}$$
$$y_{3,4} = \mu_3 + \epsilon_{3,4}$$

In an effort to write the model as $y = X\beta + \epsilon$ we will write the above as

$$y_{1,1} = 1\mu_1 + 0\mu_2 + 0\mu_3 + \epsilon_{1,1}$$
$$y_{1,2} = 1\mu_1 + 0\mu_2 + 0\mu_3 + \epsilon_{1,2}$$
$$y_{1,3} = 1\mu_1 + 0\mu_2 + 0\mu_3 + \epsilon_{1,3}$$
$$y_{1,4} = 1\mu_1 + 0\mu_2 + 0\mu_3 + \epsilon_{1,4}$$
$$y_{2,1} = 0\mu + 1\mu_2 + 0\mu_3 + \epsilon_{2,1}$$
$$y_{2,2} = 0\mu + 1\mu_2 + 0\mu_3 + \epsilon_{2,2}$$
$$y_{2,3} = 0\mu + 1\mu_2 + 0\mu_3 + \epsilon_{2,3}$$
$$y_{2,4} = 0\mu + 1\mu_2 + 0\mu_3 + \epsilon_{2,4}$$
$$y_{3,1} = 0\mu + 0\mu_2 + 1\mu_3 + \epsilon_{3,1}$$
$$y_{3,2} = 0\mu + 0\mu_2 + 1\mu_3 + \epsilon_{3,2}$$
$$y_{3,3} = 0\mu + 0\mu_2 + 1\mu_3 + \epsilon_{3,3}$$
$$y_{3,4} = 0\mu + 0\mu_2 + 1\mu_3 + \epsilon_{3,4}$$

and we will finally be able to write the matrix version

$$
\underbrace{\begin{bmatrix} y_{1,1} \\ y_{1,2} \\ y_{1,3} \\ y_{1,4} \\ y_{2,1} \\ y_{2,2} \\ y_{2,3} \\ y_{2,4} \\ y_{3,1} \\ y_{3,2} \\ y_{3,3} \\ y_{3,4} \end{bmatrix}}_{y}
=
\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{X}}
\underbrace{\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{bmatrix}}_{\beta}
+
\underbrace{\begin{bmatrix} \epsilon_{1,1} \\ \epsilon_{1,2} \\ \epsilon_{1,3} \\ \epsilon_{1,4} \\ \epsilon_{2,1} \\ \epsilon_{2,2} \\ \epsilon_{2,3} \\ \epsilon_{2,4} \\ \epsilon_{3,1} \\ \epsilon_{3,2} \\ \epsilon_{3,3} \\ \epsilon_{3,4} \end{bmatrix}}_{\epsilon}
$$

Notice that each column of the X matrix is acting as an indicator if the observation is an element of the appropriate group. As such, these are often called *indicator variables*. Another term for these, which I find less helpful, is *dummy variables*.

2.2.2.2 Offset from reference group

In this model representation of ANOVA, we have an overall mean and then offsets from the control group (which will be group one). The model is thus

$$y_{i,j} = \mu + \tau_i + \epsilon_{i,j}$$

where $\tau_1 = 0$. We can write this in matrix form as

$$
\underbrace{\begin{bmatrix} y_{1,1} \\ y_{1,2} \\ y_{1,3} \\ y_{1,4} \\ y_{2,1} \\ y_{2,2} \\ y_{2,3} \\ y_{2,4} \\ y_{3,1} \\ y_{3,2} \\ y_{3,3} \\ y_{3,4} \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}}_{\mathbf{X}} \underbrace{\begin{bmatrix} \mu \\ \tau_2 \\ \tau_3 \end{bmatrix}}_{\beta} + \underbrace{\begin{bmatrix} \epsilon_{1,1} \\ \epsilon_{1,2} \\ \epsilon_{1,3} \\ \epsilon_{1,4} \\ \epsilon_{2,1} \\ \epsilon_{2,2} \\ \epsilon_{2,3} \\ \epsilon_{2,4} \\ \epsilon_{3,1} \\ \epsilon_{3,2} \\ \epsilon_{3,3} \\ \epsilon_{3,4} \end{bmatrix}}_{\epsilon}
$$

2.3   Parameter Estimation

For both simple regression and ANOVA, we can write the model in matrix form
as

$$
y = X\beta + \epsilon \quad \text{where } \epsilon \sim N(0, \sigma^2 I_n)
$$

which could also be written as

$$
y \sim N(X\beta, \sigma^2 I_n)
$$

and we could use the maximum-likelihood principle to find estimators for β and
σ^2. In this section, we will introduce the estimators $\hat{\beta}$ and $\hat{\sigma}^2$.

2.3.1   Estimation of Location Paramters

Our goal is to find the best estimate of β given the data. To justify the formula,
consider the case where there is no error terms (i.e. $\epsilon_i = 0$ for all i). Thus we
have

$$
y = X\beta
$$

and our goal is to solve for β. To do this, we must use a matrix inverse, but
since inverses only exist for square matrices, we pre-multiple by X^T (notice that
$X^T X$ is a symmetric 2×2 matrix).

$$
X^T y = X^T X \beta
$$

and then pre-multiply by $\left(X^T X \right)^{-1}$.

$$\left(X^T X\right)^{-1} X^T y = \left(X^T X\right)^{-1} X^T X \beta$$
$$\left(X^T X\right)^{-1} X^T y = \beta$$

This exercise suggests that $\left(X^T X\right)^{-1} X^T y$ is a good place to start when looking for the maximum-likelihood estimator for β.

Happily it turns out that this quantity is in fact the maximum-likelihood estimator for the data generation model

$$y \sim N(X\beta, \sigma^2 I_n)$$

(and equivalently minimizes the sum-of-squared error). In this course we won't prove these two facts, but we will use this as our estimate of β.

$$\hat{\beta} = \left(X^T X\right)^{-1} X^T y$$

2.3.2   Estimation of Variance Parameter

Recall our simple regression model is

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

where $\epsilon_i \overset{iid}{\sim} N\left(0, \sigma^2\right)$.

Using our estimates $\hat{\beta}$ we can obtain predicted values for the regression line at any x-value. In particular we can find the predicted value for each x_i value in our dataset.

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

Using matrix notation, I would write $\hat{y} = X\hat{\beta}$.

As usual we will find estimates of the noise terms (which we will call residuals or errors) via

$$\hat{\epsilon}_i = y_i - \hat{y}_i$$
$$= y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)$$

Writing $\hat{y}$ in matrix terms we have

$$\hat{y} = X\hat{\beta}$$
$$= X\left(X^T X\right)^{-1} X^T y$$
$$= Hy$$

where $H = X \left(X^T X \right)^{-1} X^T$ is often called the *hat-matrix* because it takes y to $\hat{y}$ and has many interesting theoretical properties.[1]

We can now estimate the error terms via

$$
\begin{aligned}
\hat{\epsilon} &= y - \hat{y} \\
&= y - Hy \\
&= (I_n - H)\, y
\end{aligned}
$$

As usual we estimate σ^2 using the mean-squared error, but the general formula is

$$\hat{\sigma}^2 = \text{MSE}$$

$$= \frac{1}{n-p} \sum_{i=1}^{n} \hat{\epsilon}_i^2$$

$$= \frac{1}{n-p} \hat{\epsilon}^T \hat{\epsilon}$$

where β has p elements, and thus we have $n-p$ degrees of freedom.

2.4   Standard Errors

Because our $\hat{\beta}$ estimates vary from sample to sample, we need to estimate how much they vary from sample to sample. This will eventually allow us to create confidence intervals for and perform hypothesis test relating to the β parameter vector.

2.4.1   Expectation and variance of a random vector

Just as we needed to derive the expected value and variance of $\bar{x}$ in the previous semester, we must now do the same for $\hat{\beta}$. But to do this, we need some properties of expectations and variances.

In the following, let $A_{n \times p}$ and $b_{n \times 1}$ be constants and $\epsilon_{n \times 1}$ be a random vector.

Expectations are very similar to the scalar case where

$$
E\left[\epsilon\right] = \begin{bmatrix} E\left[\epsilon_1\right] \\ E\left[\epsilon_2\right] \\ \vdots \\ E\left[\epsilon_n\right] \end{bmatrix}
$$

---

[1]Mathematically, H is the projection matrix that takes a vector in n-dimensional space and projects it onto a p-dimension subspace spanned by the vectors in X. Projection matrices have many useful properties and much of the theory of linear models utilizes H.

and any constants are pulled through the expectation

$$E\left[A^T \epsilon + b\right] = A^T E\left[\epsilon\right] + b$$

Variances are a little different. The variance of the vector ϵ is

$$Var\left(\epsilon\right) = \begin{bmatrix} Var\left(\epsilon_1\right) & Cov\left(\epsilon_1, \epsilon_2\right) & \dots & Cov\left(\epsilon_1, \epsilon_n\right) \\ Cov\left(\epsilon_2, \epsilon_1\right) & Var\left(\epsilon_2\right) & \dots & Cov\left(\epsilon_2, \epsilon_n\right) \\ \vdots & \vdots & \ddots & \vdots \\ Cov\left(\epsilon_n, \epsilon_1\right) & Cov\left(\epsilon_n, \epsilon_2\right) & \dots & Var\left(\epsilon_1\right) \end{bmatrix}$$

and additive constants are ignored, but multiplicative constants are pulled out as follows:

$$Var\left(A^T \epsilon + b\right) = Var\left(A^T \epsilon\right) = A^T Var\left(\epsilon\right) A$$

2.4.2   Variance of Location Parameters

We next derive the sampling variance of our estimator $\hat{\beta}$ by first noting that X and β are constants and therefore

$$\begin{aligned} Var\left(y\right) &= Var\left(X\beta + \epsilon\right) \\ &= Var\left(\epsilon\right) \\ &= \sigma^2 I_n \end{aligned}$$

because the error terms are independent and therefore $Cov\left(\epsilon_i, \epsilon_j\right) = 0$ when $i \neq j$ and $Var\left(\epsilon_i\right) = \sigma^2$. Recalling that constants come out of the variance operator as the constant *squared*.

$$\begin{aligned} Var\left(\hat{\beta}\right) &= Var\left(\left(X^T X\right)^{-1} X^T y\right) \\ &= \left(X^T X\right)^{-1} X^T Var\left(y\right) X \left(X^T X\right)^{-1} \\ &= \left(X^T X\right)^{-1} X^T \sigma^2 I_n X \left(X^T X\right)^{-1} \\ &= \sigma^2 \left(X^T X\right)^{-1} X^T X \left(X^T X\right)^{-1} \\ &= \sigma^2 \left(X^T X\right)^{-1} \end{aligned}$$

Using this, the standard error (i.e. the estimated standard deviation) of $\hat{\beta}_j$ (for any j in $1, \dots, p$) is

$$StdErr\left(\hat{\beta}_j\right) = \sqrt{\hat{\sigma}^2 \left[\left(X^T X\right)^{-1}\right]_{jj}}$$

2.4.3   Summary of pertinent results

The statistic $\hat{\beta} = \left(X^T X \right)^{-1} X^T y$ is the unbiased maximum-likelihood estimator of β.

The Central Limit Theorem applies to each element of β. That is, as $n \to \infty$, the distribution of $\hat{\beta}_j \to N \left(\beta_j, \left[\sigma^2 \left(X^T X \right)^{-1} \right]_{jj} \right)$.

The error terms can be calculated via

$$\hat{y} = X\hat{\beta}$$
$$\hat{\epsilon} = y - \hat{y}$$

The estimate of σ^2 is

$$\hat{\sigma}^2 = \text{MSE} = \frac{1}{n-p} \sum_{i=1}^{n} \hat{\epsilon}_i^2 = \frac{1}{n-p} \, \hat{\epsilon}^T \hat{\epsilon}$$

and is the typical squared distance between an observation and the model prediction.

The standard error (i.e. the estimated standard deviation) of $\hat{\beta}_j$ (for any j in $1, \dots, p$) is

$$StdErr \left(\hat{\beta}_j \right) = \sqrt{ \hat{\sigma}^2 \left[\left(X^T X \right)^{-1} \right]_{jj} }$$

2.5   R example

Here we will work an example in R and do both the "hand" calculation as well as using the `lm()` function to obtain the same information.

Consider the following data in a simple regression problem:

```
n <- 20
x <- seq(0,10, length=n)
y <- -3 + 2*x + rnorm(n, sd=2)
my.data <- data.frame(x=x, y=y)
ggplot(my.data) + geom_point(aes(x=x,y=y))
```

First we must create the design matrix X. Recall

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ \vdots & \vdots \\ 1 & x_{n-1} \\ 1 & x_n \end{bmatrix}$$

and can be created in R via the following:

```
X <- cbind( rep(1,n), x)
```

Given X and y we can calculate

$$\hat{\beta} = \left(X^T X \right)^{-1} X^T y$$

in R using the following code:

```
XtXinv <- solve( t(X) %*% X )        # solve() is the inverse function
beta.hat <- XtXinv %*% t(X) %*% y %>%  # Do the calculations
  as.vector()                        # make sure the result is a vector
```

Our next step is to calculate the predicted values $\hat{y}$ and the residuals $\hat{\epsilon}$

$$\hat{y} = X\hat{\beta}$$
$$\hat{\epsilon} = y - \hat{y}$$

```
y.hat <- X %*% beta.hat
residuals <- y - y.hat
```

Now that we have the residuals, we can calculate $\hat{\sigma}^2$ and the standard errors of $\hat{\beta}_j$

$$\hat{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^{n} \hat{\epsilon}_i^2$$

$$StdErr\left(\hat{\beta}_j\right) = \sqrt{\hat{\sigma}^2 \left[\left(X^T X\right)^{-1}\right]_{jj}}$$

```r
sigma2.hat <- 1/(n-2) * sum( residuals^2 )    # p = 2
sigma.hat <- sqrt( sigma2.hat )
std.errs <- sqrt( sigma2.hat * diag(XtXinv) )
```

We now print out the important values and compare them to the summary output given by the `lm()` function in R.

```r
cbind(Est=beta.hat, StdErr=std.errs)
```

```
##          Est    StdErr
##    -1.448250 0.7438740
## x   1.681485 0.1271802
```

```r
sigma.hat
```

```
## [1] 1.726143
```

```r
model <- lm(y~x)
summary(model)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5127 -1.2134  0.1469  1.2610  3.2358
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.4482     0.7439  -1.947   0.0673 .
## x             1.6815     0.1272  13.221 1.04e-10 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.726 on 18 degrees of freedom
## Multiple R-squared:  0.9066, Adjusted R-squared:  0.9015
## F-statistic: 174.8 on 1 and 18 DF,  p-value: 1.044e-10
```

2.6 Exercises

1. We will do a simple ANOVA analysis on example 8.2 from Ott & Long-necker using the matrix representation of the model. A clinical psychologist wished to compare three methods for reducing hostility levels in university students, and used a certain test (HLT) to measure the degree of hostility. A high score on the test indicated great hostility. The psychologist used 24 students who obtained high and nearly equal scores in the experiment. eight were selected at random from among the 24 problem cases and were treated with method 1. Seven of the remaining 16 students were selected at random and treated with method 2. The remaining nine students were treated with method 3. All treatments were continued for a one-semester period. Each student was given the HLT test at the end of the semester, with the results show in the following table. (This analysis was done in section 8.3 of my STA 570 notes)

Method	Values
1	96, 79, 91, 85, 83, 91, 82, 87
2	77, 76, 74, 73, 78, 71, 80
3	66, 73, 69, 66, 77, 73, 71, 70, 74

We will be using the cell means model of ANOVA

$$y_{ij} = \beta_i + \epsilon_{ij}$$

where β_i is the mean of group i and $\epsilon_{ij} \overset{iid}{\sim} N\left(0, \sigma^2\right)$.

   a. Create one vector of all 24 hostility test scores y. (Use the c() function.)

   b. Create a design matrix X with dummy variables for columns that code for what group an observation belongs to. Notice that X will be a 24 rows by 3 column matrix. *Hint: An R function that might be handy is cbind(a,b) which will bind two vectors or matrices together along the columns. There is also a corresponding rbind() function that binds vectors/matrices along rows. Furthermore, the repeat command rep() could be handy.*

c) Find $\hat{\beta}$ using the matrix formula given in class. *Hint: The R function* `t(A)` *computes the matrix transpose* $\mathbf{A}^T$, `solve(A)` *computes* $\mathbf{A}^{-1}$, *and the operator* `%*%` *does matrix multiplication (used as* `A %*% B`*).*

d) Examine the matrix $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$. What do you notice about it? In particular, think about the result when you right multiply by $\mathbf{y}$. How does this matrix calculate the appropriate group means and using the appropriate group sizes n_i?

2. We will calculate the y-intercept and slope estimates in a simple linear model using matrix notation. We will use a data set that gives the diameter at breast height (DBH) versus tree height for a randomly selected set of trees. In addition, for each tree, a ground measurement of crown closure (CC) was taken. Larger values of crown closure indicate more shading and is often associated with taller tree morphology (possibly). We will be interested in creating a regression model that predicts height based on DBH and CC. In the interest of reduced copying, we will only use 10 observations. *(Note: I made this data up and the DBH values might be unrealistic. Don't make fun of me.)*

DBH	30.5	31.5	31.7	32.3	33.3	35	35.4	35.6	36.3	37.8
CC	0.74	0.69	0.65	0.72	0.58	0.5	0.6	0.7	0.52	0.6
Height	58	64	65	70	68	63	78	80	74	76

We are interested in fitting the regression model

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \epsilon_i$$

where β_0 is the y-intercept and β_1 is the slope parameter associated with DBH and β_2 is the slope parameter associated with Crown Closure.

a) Create a vector of all 10 heights $\mathbf{y}$.
b) Create the design matrix $\mathbf{X}$.
c) Find $\hat{\beta}$ using the matrix formula given in class.
d) Compare your results to the estimated coefficients you get using the `lm()` function. To add the second predictor to the model, your call to `lm()` should look something like `lm(Height ~ DBH + CrownClosure)`.

Chapter 3

Inference

```r
library(tidymodels) # Grab model results as data frames
library(tidyverse)  # ggplot2, dplyr, tidyr
```

Learning Outcomes

- Utilize $\hat{\beta}$ and its standard errors of to produce confidence intervals and hypothesis tests for β_j values.
- Convert hypothesis tests for $\beta_j = 0$ into a model comparison F-test.
- Utilize F-tests to perform a hypothesis test of multiple β_j values all being equal to zero. This is the simple vs complex model comparison.
- Create confidence intervals for β_j values leveraging the normality assumption of residuals.

3.1 Introduction

The goal of statistics is to take information calculated from sample data and use that information to estimate population parameters. The problem is that the sample statistic is only a rough guess and if we were to collect another sample of data, we'd get a different sample statistic and thus a different parameter estimate. Therefore, we need to utilize the sample statistics to create confidence intervals and make hypothesis tests about those parameters.

In this chapter, we'll consider a dataset about the Galápagos Islands relating the number of tortoise species on an island to various island characteristics such as size, maximum elevation, etc. The set contains $n = 30$ islands and

Variable	Description
Species	Number of tortoise species found on the island
Endimics	Number of tortoise species endemic to the island
Elevation	Elevation of the highest point on the island
Area	Area of the island (km^2)
Nearest	Distance to the nearest neighboring island (km)
Scruz	Distance to the Santa Cruz islands (km)
Adjacent	Area of the nearest adjacent island (km^2)

```
data('gala', package='faraway')     # import the data set
head(gala)                          # show the first couple of rows
```

```
##               Species Endemics  Area Elevation Nearest Scruz Adjacent
## Baltra             58       23 25.09       346     0.6   0.6     1.84
## Bartolome          31       21  1.24       109     0.6  26.3   572.33
## Caldwell            3        3  0.21       114     2.8  58.7     0.78
## Champion           25        9  0.10        46     1.9  47.4     0.18
## Coamano             2        1  0.05        77     1.9   1.9   903.82
## Daphne.Major       18       11  0.34       119     8.0   8.0     1.84
```

3.2 Confidence Intervals and Hypothesis Tests

We can now state the general method of creating confidence intervals and perform hypothesis tests for any element of β.

The general recipe for a $(1 - \alpha) * 100\%$ confidence interval is

$$\text{Estimate} \pm Q^*_{1-\alpha/2} \, \text{StdErr}(\text{Estimate})$$

where $Q^*_{1-\alpha/2}$ is the $1 - \alpha/2$ quantile from some appropriate distribution. The mathematical details about which distribution the quantile should come from are often obscure, but usually involve the degrees of freedom $n - p$ where p is the number of parameters in the "signal" part of the model.

The confidence interval formula for the β parameters in a linear model is

$$\hat{\beta}_j \pm t^*_{1-\alpha/2,n-p} \, StdErr\left(\hat{\beta}_j\right)$$

where $t^*_{1-\alpha/2,n-p}$ is the $1-\alpha/2$ quantile from the t-distribution with $n-p$ degrees of freedom. A test statistic for testing $H_0 : \beta_j = 0$ versus $H_a : \beta_j \neq 0$ is

$$t_{n-p} = \frac{\hat{\beta}_j - 0}{StdErr\left(\hat{\beta}_j\right)}$$

3.3 F-tests

We wish to develop a rigorous way to compare nested models and decide if a complicated model explains enough more variability than a simple model to justify the additional intellectual effort of thinking about the data in the complicated fashion.

It is important to specify that we are developing a way of testing nested models. By nested, we mean that the simple model can be created from the full model just by setting one or more model parameters to zero. This method doesn't constrain us for testing just a single parameter being possibly zero. Instead we can test if an entire set of parameters all possibly being equal to zero.

3.3.1 Theory

Recall that in the simple regression and ANOVA cases we were interested in comparing a simple model versus a more complex model. For each model we computed the sum of squares error (SSE) and said that if the complicated model performed much better than the simple then $SSE_{simple} \gg SSE_{complex}$.

Recall from the estimation chapter, the model parameter estimates are found by using the $\hat{\beta}$ values that minimize the SSE. If it were to turn out that a $\hat{\beta}_j$ of zero minimized SSE, then zero would be estimate. Next consider that we are requiring the simple model to be a simplification of the complex model by setting certain parameters to zero. So we are considering a simple model that sets $\hat{\beta}_j = 0$ and vs a complex model that allows for $\hat{\beta}_j$ to be any real value (including), then because we select $\hat{\beta}_j$ to be the value that minimizes SSE, then $SSE_{simple} \geq SSE_{complex}$.

We'll define $SSE_{difference} = SSE_{simple} - SSEcomplex \geq 0$ and observe that if the complex model is a much better fit to the data, then $SSE_{difference}$ is large. But how large is large enough to be statistically significant? In part, it depends on how many more parameters were added to the model and what the amount of unexplained variability left in the complex model. Let df_{diff} be the number of parameters difference between the simple and complex models.

As with most test statistics, the F statistic can be considered as a "Signal-to-Noise" ratio where the signal part is the increased amount of variability explained per additional parameter by the complex model and the noise part is just the MSE of the complex model.

$$F = \frac{\text{Signal}}{\text{Noise}} = \frac{RSS_{difference}/df_{diff}}{RSS_{complex}/df_{complex}}$$

and we claimed that if the null hypothesis was true (i.e. the complex model is an unnecessary obfuscation of the simple), then this ratio follows an F-distribution with degrees of freedom df_{diff} and $df_{complex}$.

The F-distribution is centered near one and we should reject the simple model (in favor of the complex model) if this F statistic is much larger than one. Therefore the p-value for the test is

$F_{df_{diff}, \ df_{complex}}$ distribution
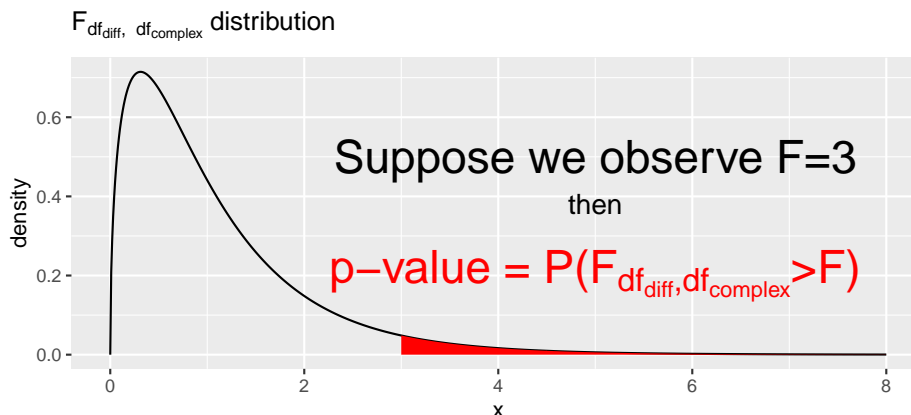
This hypothesis test doesn't require a particular difference in number of parameters in each model, while the single parameter t-test is stuck testing if just a single parameter is possibly zero. In the single parameter test case, the t-test and F-test give the same prior hypothesis test previous t-test. The a corresponding t-test and F-test will give the same p-value and therefore the same inference about if β_j is possibly zero.

3.4   Example

We will consider a data set from Johnson and Raven (1973) which also appears in Weisberg (1985). This data set is concerned with the number of tortoise species on $n = 30$ different islands in the Galapagos. The variables of interest in the data set are:

Variable	Description
Species	Number of tortoise species found on the island
Endimics	Number of tortoise species endemic to the island
Elevation	Elevation of the highest point on the island
Area	Area of the island (km^2)
Nearest	Distance to the nearest neighboring island (km)
Scruz	Distance to the Santa Cruz islands (km)
Adjacent	Area of the nearest adjacent island (km^2)

We will first read in the data set from the package `faraway`.

```
data('gala', package='faraway')      # import the data set
head(gala)                           # show the first couple of rows
```

```
##              Species Endemics  Area Elevation Nearest Scruz Adjacent
## Baltra            58       23 25.09       346     0.6   0.6     1.84
## Bartolome         31       21  1.24       109     0.6  26.3   572.33
## Caldwell           3        3  0.21       114     2.8  58.7     0.78
## Champion          25        9  0.10        46     1.9  47.4     0.18
## Coamano            2        1  0.05        77     1.9   1.9   903.82
## Daphne.Major      18       11  0.34       119     8.0   8.0     1.84
```

First we will create the full model that predicts the number of species as a function of elevation, area, nearest, scruz and adjacent. Notice that this model has $p = 6$ β_i values (one for each coefficient plus the intercept).

$$y_i = \beta_0 + \beta_1 Area_i + \beta_2 Elevation_i + \beta_3 Nearest_i + \beta_4 Scruz_i + \beta_5 Adjacent_i + \epsilon_i$$

We can happily fit this model just by adding terms on the left hand side of the model formula. Notice that R creates the design matrix X for us.

```
M.c <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent, data=gala)
model.matrix(M.c)   # this is the design matrix X.
```

```
##              (Intercept)    Area Elevation Nearest Scruz Adjacent
## Baltra                 1   25.09       346     0.6   0.6     1.84
## Bartolome              1    1.24       109     0.6  26.3   572.33
## Caldwell               1    0.21       114     2.8  58.7     0.78
## Champion               1    0.10        46     1.9  47.4     0.18
## Coamano                1    0.05        77     1.9   1.9   903.82
## Daphne.Major           1    0.34       119     8.0   8.0     1.84
## Daphne.Minor           1    0.08        93     6.0  12.0     0.34
## Darwin                 1    2.33       168    34.1 290.2     2.85
## Eden                   1    0.03        71     0.4   0.4    17.95
## Enderby                1    0.18       112     2.6  50.2     0.10
## Espanola               1   58.27       198     1.1  88.3     0.57
## Fernandina             1  634.49      1494     4.3  95.3  4669.32
## Gardner1               1    0.57        49     1.1  93.1    58.27
## Gardner2               1    0.78       227     4.6  62.2     0.21
## Genovesa               1   17.35        76    47.4  92.2   129.49
## Isabela                1 4669.32      1707     0.7  28.1   634.49
## Marchena               1  129.49       343    29.1  85.9    59.56
## Onslow                 1    0.01        25     3.3  45.9     0.10
```

```
## Pinta                      1    59.56      777    29.1 119.6    129.49
## Pinzon                     1    17.95      458    10.7  10.7      0.03
## Las.Plazas                 1     0.23       94     0.5   0.6     25.09
## Rabida                     1     4.89      367     4.4  24.4    572.33
## SanCristobal              1   551.62      716    45.2  66.6      0.57
## SanSalvador               1   572.33      906     0.2  19.8      4.89
## SantaCruz                 1   903.82      864     0.6   0.0      0.52
## SantaFe                   1    24.08      259    16.5  16.5      0.52
## SantaMaria                1   170.92      640     2.6  49.2      0.10
## Seymour                   1     1.84      147     0.6   9.6     25.09
## Tortuga                   1     1.24      186     6.8  50.9     17.95
## Wolf                      1     2.85      253    34.1 254.7      2.33
## attr(,"assign")
## [1] 0 1 2 3 4 5
```

All the usual calculations from chapter two can be calculated and we can see
the summary table for this regression as follows:

```
summary(M.c)
```

```
##
## Call:
## lm(formula = Species ~ Area + Elevation + Nearest + Scruz + Adjacent,
##     data = gala)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -111.679  -34.898   -7.862   33.460  182.584
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.068221  19.154198   0.369 0.715351
## Area        -0.023938   0.022422  -1.068 0.296318
## Elevation    0.319465   0.053663   5.953 3.82e-06 ***
## Nearest      0.009144   1.054136   0.009 0.993151
## Scruz       -0.240524   0.215402  -1.117 0.275208
## Adjacent    -0.074805   0.017700  -4.226 0.000297 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.98 on 24 degrees of freedom
## Multiple R-squared:  0.7658, Adjusted R-squared:  0.7171
## F-statistic:  15.7 on 5 and 24 DF,  p-value: 6.838e-07
```

3.4.1   Testing All Covariates

The first test we might want to do is to test if any of the covariates are significant. That is to say that we want to test the full model versus the simple null hypothesis model

$$y_i = \beta_0 + \epsilon_i$$

that has no covariates and only a y-intercept. So we will create a simple model

```
M.s <- lm(Species ~ 1, data=gala)
```

and calculate the appropriate Residual Sums of Squares (RSS) for each model, along with the difference in degrees of freedom between the two models.

```
RSS.c <- sum(resid(M.c)^2)
RSS.s <- sum(resid(M.s)^2)
df.diff <- 5                  # complex model has 5 additional parameters
df.c <- 30 - 6                # complex model has 24 degrees of freedom left
```

The F-statistic for this test is therefore

```
F.stat <-  ( (RSS.s - RSS.c) / df.diff ) / ( RSS.c / df.c )
F.stat
```

```
## [1] 15.69941
```

and should be compared against the F-distribution with 5 and 24 degrees of freedom. Because a large difference between RSS.s and RSS.c would be evidence for the alternative, larger model, the p-value for this test is

$$p-value = P\left(F_{5,24} \geq \texttt{F.stat}\right)$$

```
p.value <-  1 - pf(15.699, 5, 24)
p.value
```

```
## [1] 6.839486e-07
```

Both the F.stat and its p-value are given at the bottom of the summary table. However, I might be interested in creating an ANOVA table for this situation.

Source	df	Sum Sq	Mean Sq	F	p-value
Difference	$p-1$	RSS_d	$MSE_d = RSS_d/(p-1)$	MSE_d/MSE_c	$P(F > F_{p-1,n-p})$
Complex	$n-p$	RSS_c	$MSE_c = RSS_c/(n-p)$		
Simple	$n-1$	RSS_s			

This type of table is often shown in textbooks, but base functions in R don't produce exactly this table. Instead the `anova(simple, complex)` command produces the following:

Models	df	RSS	Diff in RSS	F	p-value
Simple	$n-1$	RSS_s			
Complex	$n-p$	RSS_c	RSS_d	MSE_d/MSE_c	$P(F > F_{p-1,n-p})$

can be obtained from R by using the `anova()` function on the two models of interest. This representation skips showing the Mean Squared calculations.

```
anova(M.s, M.c)
```

```
## Analysis of Variance Table
##
## Model 1: Species ~ 1
## Model 2: Species ~ Area + Elevation + Nearest + Scruz + Adjacent
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     29 381081
## 2     24  89231  5    291850 15.699 6.838e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.4.2   Testing a Single Covariate

For a particular covariate, β_j, we might wish to perform a test to see if it can be removed from the model. It can be shown that the F-statistic can be re-written as

$$F = \frac{[RSS_s - RSS_c]/1}{RSS_c/(n-p)}$$

$$=:$$

$$= \left[\frac{\hat{\beta}_j}{SE\left(\hat{\beta}_j\right)}\right]^2$$

$$= t^2$$

where t has a t-distribution with $n-p$ degrees of freedom under the null hypothesis that the simple model is sufficient.

We consider the case of removing the covariate `Area` from the model and will calculate our test statistic using both methods.

```
M.c <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent, data=gala)
M.s <- lm(Species ~        Elevation + Nearest + Scruz + Adjacent, data=gala)
RSS.c <- sum( resid(M.c)^2 )
RSS.s <- sum( resid(M.s)^2 )
df.d <- 1
df.c <- 30-6
F.stat <- ((RSS.s - RSS.c)/1) / (RSS.c / df.c)
F.stat
```

```
## [1] 1.139792
```

```
1 - pf(F.stat, 1, 24)
```

```
## [1] 0.296318
```

```
sqrt(F.stat)
```

```
## [1] 1.067611
```

To calculate it using the estimated coefficient and its standard error, we must grab those values from the summary table

```
broom::tidy(M.c)   # get the coefficient table
```

```
## # A tibble: 6 x 5
##   term          estimate std.error statistic   p.value
##   <chr>            <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)  7.07        19.2       0.369   0.715
## 2 Area        -0.0239       0.0224   -1.07    0.296
## 3 Elevation    0.319        0.0537    5.95    0.00000382
## 4 Nearest      0.00914      1.05      0.00867 0.993
## 5 Scruz       -0.241        0.215    -1.12    0.275
## 6 Adjacent    -0.0748       0.0177   -4.23    0.000297
```

```
beta.area    <- broom::tidy(M.c)[2,2] %>% pull()   # pull turns it into a scalar
SE.beta.area <- broom::tidy(M.c)[2,3] %>% pull()
t <- beta.area / SE.beta.area
t
```

```
## [1] -1.067611
```

```
2 * pt(t, 24)
```

```
## [1] 0.296318
```

All that hand calculation is tedious, so we can again use the `anova()()` command to compare the two models.

```
anova(M.s, M.c)
```

```
## Analysis of Variance Table
##
## Model 1: Species ~ Elevation + Nearest + Scruz + Adjacent
## Model 2: Species ~ Area + Elevation + Nearest + Scruz + Adjacent
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     25 93469
## 2     24 89231  1    4237.7 1.1398 0.2963
```

3.4.3   Testing a Subset of Covariates

Often a researcher will want to remove a subset of covariates from the model. In the Galapagos example, Area, Nearest, and Scruz all have non-significant p-values and would be removed when comparing the full model to the model without that one covariate. While each of them might be non-significant, is the sum of all three significant?

Because the individual $\hat{\beta}_j$ values are not independent, then we cannot claim that the subset is not statistically significant just because each variable in turn was insignificant. Instead we again create simple and complex models in the same fashion as we have previously done.

```
M.c <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent, data=gala)
M.s <- lm(Species ~        Elevation +                    Adjacent, data=gala)
anova(M.s, M.c)
```

```
## Analysis of Variance Table
##
## Model 1: Species ~ Elevation + Adjacent
## Model 2: Species ~ Area + Elevation + Nearest + Scruz + Adjacent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     27 100003
## 2     24  89231  3     10772 0.9657  0.425
```

We find a large p-value associated with this test and can safely stay with the null hypothesis, that the simple model is sufficient to explain the observed variability in the number of species of tortoise.

3.5 Exercises

1. The dataset prostate in package **faraway** has information about a study of 97 men with prostate cancer. We import the data and examine the first four observations using the following commands.

```
data(prostate, package='faraway')
head(prostate)
```

It is possible to get information about the data set using the command `help(prostate)`. Fit a model with `lpsa` as the response and all the other variables as predictors.

   a) Compute 90% and 95% confidence intervals for the parameter associated with `age`. Using just these intervals, what could we deduced about the p-value for age in the regression summary. *Hint: look at the help for the function* `confint()`. *You'll find the* `level` *option to be helpful. Alternatively use the* `broom::tidy()` *function with the* `conf.int=TRUE` *option and also use the* `level=` *option as well.*

   b) Remove all the predictors that are not significant at the 5% level. Test this model against the original model. Which is preferred?

2. Thirty samples of cheddar cheese were analyzed for their content of acetic acid, hydrogen sulfide and lactic acid. Each sample was tasted and scored by a panel of judges and the average taste score produces. Used the **cheddar** dataset from the **faraway** package (import it the same way you did in problem one, but now use **cheddar**) to answer the following:

   a) Fit a regression model with taste as the response and the three chemical contents as predictors. Identify the predictors that are statistically significant at the 5% level.

   b) `Acetic` and `H2S` are measured on a $\log_{10}$ scale. Create two new columns in the **cheddar** data frame that contain the values on their original scale. Fit a linear model that uses the three covariates on their non-log scale. Identify the predictors that are statistically significant at the 5% level for this model.

   c) Can we use an F-test to compare these two models? Explain why or why not. Which model provides a better fit to the data? Explain your reasoning.

   d) For the model in part (a), if a sample of cheese were to have `H2S` increased by 2 (where H2S is on the log scale and we increase this value by 2 using some method), what change in taste would be expected? What caveates must be made in this interpretation? *Hint: I don't want to get into interpreting parameters on the log scale just yet. So just interpret this as adding 2 to the covariate value and predicting the change in taste.*

3. The `sat` data set in the `faraway` package gives data collected to study the relationship between expenditures on public education and test results.

   a) Fit a model that with `total` SAT score as the response and only the intercept as a covariate.

   b) Fit a model with `total` SAT score as the response and `expend`, `ratio`, and `salary` as predictors (along with the intercept).

   c) Compare the models in parts (a) and (b) using an F-test. Is the larger model superior?

   d) Examine the summary table of the larger model? Does this contradict your results in part (c)? What might be causing this issue? Create a graph or summary diagnostics to support your guess.

   e) Fit the model with `salary` and `ratio` (along with the intercept) as predictor variables and examine the summary table. Which covariates are significant?

   f) Now add `takers` to the model (so the model now includes three predictor variables along with the intercept). Test the hypothesis that $\beta_{takers} = 0$ using the summary table.

   g) Discuss why `ratio` was not significant in the model in part (e) but was significant in part (f). *Hint: Look at the Residual Standard Error $\hat{\sigma}$ in each model and argue that each t-statistic is some variant of a "signal-to-noise" ratio and that the "noise" part is reduced in the second model.*

4. In this exercise, we will show that adding a covariate to the model that is just random noise will decrease the model Sum of Squared Error (SSE).

   a) Fit a linear model to the `trees` dataset that is always preloaded in R. Recall that this dataset has observations from 31 cherry trees with variables tree height, girth and volume of lumber produced. Fit Volume ~ Height.

   b) From this simple regression model, obtain the SSE. *Hint: you can calculate this yourself, pull it from the broom::glance() output where it is entitled **deviance** or extract it from the output of the **anova()** command.*

   c) Add a new covariate to the model named `Noise` that is generated at random from a uniform distribution using the following code:

   ```
   trees <- trees %>%
     mutate( Noise = runif( n() ) )
   ```

   d) Fit a linear model that includes this new `Noise` variable in addition to the `Height`. Calculate the SSE error in the same manner as before. Does it decrease or increase. Quantify how much it has changed.

e) Repeat parts (c) and (d) several times. Comment on the trend in change in SSE. *Hint: This isn't strictly necessary but is how I would go about answering this question. Wrap parts (c) and (d) in a `for` loop and generate a data.frame of a couple hundred runs. Then make a density plot of the SSE values for the complex models and add a vertical line on the graph of the simple model SSE.*

```
results <- NULL
for( i in 1:2000 ){
  # Do stuff
  results <- results %>% rbind( glance(model) )
}
ggplot(results, aes(x=deviance)) +
  geom_density() +
  geom_vline( xintercept = simple.SSE )
```

Chapter 4

Contrasts

```r
library(tidyverse)    # ggplot2, dplyr, tidyr
library(tidymodels)   # for broom functions
library(emmeans)      # for emmeans()
# library(multcomp)   # for glht()  - multcomp fights with dplyr, :(
```

4.1 Introduction

We have written our model as $y = X\beta + \epsilon$ and often we are interested in some linear function of the $\hat{\beta}$. Some examples include the model predictions $\hat{y}_i = X_{i.}\beta$ where $X_{i.}$ is the ith row of the X matrix. Other examples include differences in group means in a one-way ANOVA or differences in predicted values $\hat{y}_i - \hat{y}_j$.

All of these can can be written as $c'\beta$ for some vector c.

We often are interested in estimating a function of the parameters β. For example in the offset representation of the ANOVA model with 3 groups we have

$$y_{ij} = \mu + \tau_i + \epsilon_{ij}$$

where

$$\beta = \begin{bmatrix} \mu\ \tau_2\ \tau_3 \end{bmatrix}^T$$

and μ is the mean of the control group, group one is the control group and thus $\tau_1 = 0$, and τ_2 and τ_3 are the offsets of group two and three from the control group. In this representation, the mean of group two is $\mu + \tau_2$ and is estimated with $\hat{\mu} + \hat{\tau}_2$.

A contrast is a linear combinations of elements of $\hat{\beta}$, which is a fancy way of saying that it is a function of the elements of $\hat{\beta}$ where the elements can be

49

added, subtracted, or multiplied by constants. In particular, the contrast can be represented by the vector c such that the function we are interested in is $c^T \hat{\beta}$.

In the ANOVA case with $k = 3$ where we have the offset representation, I might be interested in the mean of group 2, which could be written as

$$\hat{\mu}_2 = \hat{\mu} + \hat{\tau}_2 = \underbrace{\left[\begin{array}{ccc} 1 & 1 & 0 \end{array}\right]}_{c^T} \cdot \underbrace{\left[\begin{array}{c} \hat{\mu} \\ \hat{\tau}_2 \\ \hat{\tau}_3 \end{array}\right]}_{\hat{\beta}}$$

Similarly in the simple regression case, I will be interested in the height of the regression line at x_0. This height can be written as

$$\hat{y}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0 = \underbrace{\left[\begin{array}{cc} 1 & x_0 \end{array}\right]}_{c^T} \cdot \underbrace{\left[\begin{array}{c} \hat{\beta}_0 \\ \hat{\beta}_1 \end{array}\right]}_{\hat{\beta}}$$

In this manner, we could think of calculating all of the predicted values $\hat{y}_i$ as just the result of the contrast $X\hat{\beta}$ where our design matrix X takes the role of the contrasts.

4.2   Estimate and variance

One of the properties of maximum likelihood estimator (MLEs), is that they are invariant under transformations. Meaning that since $\hat{\beta}$ is the MLE of β, then $c^T \hat{\beta}$ is the MLE of $c^T \beta$. The only thing we need to perform hypotheses tests and create confidence intervals is an estimate of the variance of $c^T \hat{\beta}$.

Because we know the variance of $\hat{\beta}$ is

$$Var\left(\hat{\beta}\right) = \sigma^2 \left(X^T X\right)^{-1}$$

and because c is a constant, then

$$Var\left(c^T \hat{\beta}\right) = \sigma^2 c^T \left(X^T X\right)^{-1} c$$

and the standard error is found by plugging in our estimate of σ^2 and taking the square root.

$$StdErr\left(c^T \hat{\beta}\right) = \sqrt{\hat{\sigma}^2 c^T \left(X^T X\right)^{-1} c} = \hat{\sigma}\sqrt{c^T \left(X^T X\right)^{-1} c}$$

As usual, we can now calculate confidence intervals for $c^T \hat{\beta}$ using the usual formula

$$Est \pm t_{n-p}^{1-\alpha/2} \; StdErr\left(Est\right)$$

$$c^T \hat{\beta} \pm t_{n-p}^{1-\alpha/2} \, \hat{\sigma} \sqrt{c^T \left(X^T X \right)^{-1} c}$$

Recall the hostility example which was an ANOVA with three groups with the data

Method	Test Scores
1	96 79 91 85 83 91 82 87
2	77 76 74 73 78 71 80
3	66 73 69 66 77 73 71 70 74

We have analyzed this data using both the cell means model and the offset and we will demonstrate how to calculate the group means from the offset representation. Thus we are interested in estimating $\mu + \tau_2$ and $\mu + \tau_3$. I am also interested in estimating the difference between treatment 2 and 3 and will therefore be interested in estimating $\tau_2 - \tau_3$.

```
data <- data.frame(
  y =   c(96,79,91,85,83,91,82,87,
          77,76,74,73,78,71,80,
          66,73,69,66,77,73,71,70,74),
  group = factor(c( rep('Group1',8), rep('Group2',7),rep('Group3',9) ))
)

ggplot(data, aes(x=group, y=y)) + geom_boxplot()
```
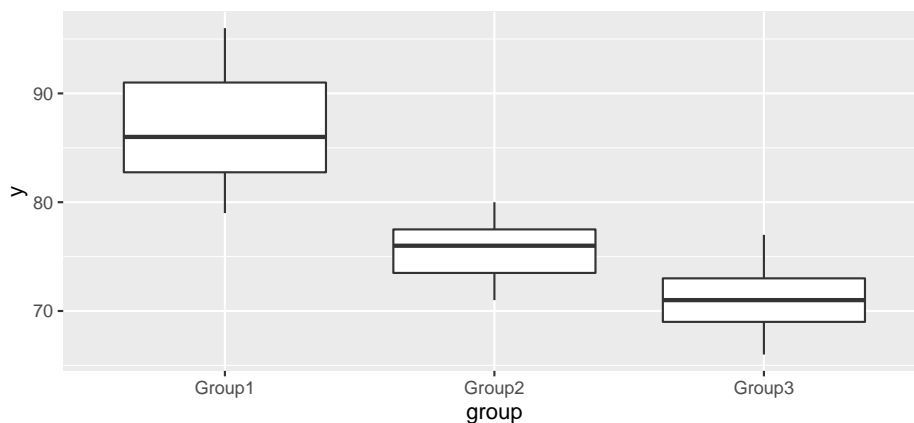

We can fit the offset model and obtain the design matrix and estimate of $\hat{\sigma}$ via the following code.

```r
m <- lm(y ~ group, data=data)          # Fit the ANOVA model (offset representation)
tidy(m)                                # Show me beta.hat
```

```
## # A tibble: 3 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)      86.8      1.52      57.1  1.57e-24
## 2 groupGroup2     -11.2      2.22     -5.03 5.58e- 5
## 3 groupGroup3     -15.8      2.09     -7.55 2.06e- 7
```

```r
X <- model.matrix(m)                        # obtains the design matrix
sigma.hat <- glance(m) %>% pull(sigma) # grab sigma.hat
beta.hat <- tidy(m) %>% pull(estimate)
XtX.inv <- solve( t(X) %*% X )
```

Now we calculate

```r
contr <- c(1,1,0)   # define my contrast
ctb <- t(contr) %*% beta.hat
std.err <- sigma.hat * sqrt( t(contr) %*% XtX.inv %*% contr )

data.frame(Estimate=ctb, StdErr=std.err)
```

```
##   Estimate   StdErr
## 1 75.57143 1.622994
```

and notice this is the exact same estimate and standard error we got for group two when we fit the cell means model.

```r
CellMeansModel <- lm(y ~ group - 1, data=data)
CellMeansModel %>% tidy()
```

```
## # A tibble: 3 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 groupGroup1      86.7      1.52      57.1 1.57e-24
## 2 groupGroup2      75.6      1.62      46.6 1.12e-22
## 3 groupGroup3      71        1.43      49.6 2.99e-23
```

4.3   Estimating contrasts using `glht()`

Instead of us doing all the matrix calculations ourselves, all we really need is to is specify the row vector c^T. The function that will do the rest of the calculations

is the generalized linear hypothesis test function `glht()` that can be found in the multiple comparisons package `multcomp`. The p-values will be adjusted to correct for testing multiple hypothesis, so there may be slight differences compared to the p-value seen in just the regular summary table.

4.3.1 1-way ANOVA

We will again use the Hostility data set and demonstrate how to calculate the point estimates, standard errors and confidence intervals for the group means given a model fit using the offset representation.

```
m <- lm(y ~ group, data=data)
m %>% tidy()
```

```
## # A tibble: 3 x 5
##    term        estimate std.error statistic  p.value
##    <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     86.8      1.52     57.1  1.57e-24
## 2 groupGroup2    -11.2      2.22     -5.03 5.58e- 5
## 3 groupGroup3    -15.8      2.09     -7.55 2.06e- 7
```

We will now define a row vector (and it needs to be a matrix or else `glht()` will throw an error. First we note that the simple contrast $c^T = [1\ 0\ 0]$ just grabs the first coefficient and gives us the same estimate and standard error as the summary did.

```
contr <- rbind("Intercept"=c(1,0,0))       # 1x3 matrix with row named "Intercept"
test <- multcomp::glht(m, linfct=contr)    # the linear function to be tested is contr
summary(test)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = y ~ group, data = data)
##
## Linear Hypotheses:
##                Estimate Std. Error t value Pr(>|t|)
## Intercept == 0   86.750      1.518   57.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Next we calculate the estimate of all the group means μ, $\mu + \tau_2$ and $\mu + \tau_3$ and the difference between group 2 and 3. Notice I can specify more than one contrast at a time.

```r
contr <- rbind("Mean of Group 1"=c(1,0,0),
               "Mean of Group 2"=c(1,1,0),
               "Mean of Group 3"=c(1,0,1),
               "Diff G2-G3"     =c(0,1,-1))
test <- multcomp::glht(m, linfct=contr)
summary(test)
```

```
##
##    Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = y ~ group, data = data)
##
## Linear Hypotheses:
##                    Estimate Std. Error t value Pr(>|t|)
## Mean of Group 1 == 0   86.750      1.518  57.141   <0.001 ***
## Mean of Group 2 == 0   75.571      1.623  46.563   <0.001 ***
## Mean of Group 3 == 0   71.000      1.431  49.604   <0.001 ***
## Diff G2-G3 == 0         4.571      2.164   2.112    0.144
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Finally we calculate confidence intervals in the usual manner using the
confint() function.

```r
confint(test, level=0.95)
```

```
##
##    Simultaneous Confidence Intervals
##
## Fit: lm(formula = y ~ group, data = data)
##
## Quantile = 2.6461
## 95% family-wise confidence level
##
##
## Linear Hypotheses:
##                    Estimate lwr       upr
## Mean of Group 1 == 0 86.7500  82.7328 90.7672
## Mean of Group 2 == 0 75.5714  71.2769 79.8660
## Mean of Group 3 == 0 71.0000  67.2126 74.7874
## Diff G2-G3 == 0        4.5714  -1.1546 10.2975
```

4.4   Using `emmeans` Package

Specifying the contrasts by hand is extremely difficult to do correctly and instead we would prefer to specify the contrasts using language like "create all possible pairwise contrasts" where each pair is just a subtraction. The R-package `emmeans` tries to simply the creation of common contrasts.

To show how to use the `emmeans` package, we'll consider a bunch of common models and show how to address common statistical questions for each.

4.4.1   Simple Regression

There is a dataset built into R named `trees` which describes a set of $n = 31$ cherry trees and the goal is to predict the volume of timber produced by each tree just using the tree girth a 4.5 feet above the ground.

```r
data(trees)
model <- lm( Volume ~ Girth, data=trees )
trees <- trees %>%
  dplyr::select( -matches('fit'), -matches('lwr'), -matches('upr') ) %>%
  cbind( predict(model, interval='conf'))
ggplot(trees, aes(x=Girth, y=Volume)) +
  geom_point() +
  geom_ribbon( aes(ymin=lwr, ymax=upr), alpha=.3 ) +
  geom_line( aes(y=fit) )
```


Using the `summary()` function, we can test hypotheses about if the y-intercept or slope could be equal to zero, but we might be interested in confidence intervals for the regression line at girth values of 10 and 12.

```r
# We could find the regression line heights and CI using
# either predict() or emmeans()
predict(model, newdata=data.frame(Girth=c(10,12)), interval='conf' )
```

```
##         fit      lwr      upr
## 1 13.71511 11.44781 15.98240
## 2 23.84682 22.16204 25.53159
```

```r
emmeans(model, specs = ~Girth, at=list(Girth=c(10,12)) )
```

```
##  Girth emmean    SE df lower.CL upper.CL
##     10   13.7 1.109 29     11.4     16.0
##     12   23.8 0.824 29     22.2     25.5
##
## Confidence level used: 0.95
```

The `emmeans()` function requires us to specify the grid of reference points we are interested as well as which variable or variables we wish to separate out. In the simple regression case, the `specs` argument is just the single covariate.

We might next ask if the difference in volume between a tree with 10 inch girth is statistically different than a tree with 12 inch girth? In other words, we want to test

$$H_0 : \ (\beta_0 + \beta_1 \cdot 10) - (\beta_0 + \beta_1 \cdot 12) = 0$$
$$H_a : \ (\beta_0 + \beta_1 \cdot 10) - (\beta_0 + \beta_1 \cdot 12) \neq 0$$

In this case, we want to look at all possible pairwise differences between the predicted values at 10 and 12. (The *rev* part just reverses the order in which we do the subtraction.)

```r
emmeans(model, specs = revpairwise~Girth,
        at=list(Girth=c(10,12)) )
```

```
## $emmeans
##  Girth emmean    SE df lower.CL upper.CL
##     10   13.7 1.109 29     11.4     16.0
##     12   23.8 0.824 29     22.2     25.5
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  12 - 10      10.1 0.495 29  20.478  <.0001
```

Notice that if I was interested in 3 points, we would get all of the differences.

```r
emmeans(model, specs = pairwise~Girth,
        at=list(Girth=c(10,11,12)) )
```

```
## $emmeans
##  Girth emmean    SE df lower.CL upper.CL
##    10   13.7 1.109 29    11.4     16.0
##    11   18.8 0.945 29    16.8     20.7
##    12   23.8 0.824 29    22.2     25.5
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  10 - 11    -5.07 0.247 29 -20.478 <.0001
##  10 - 12   -10.13 0.495 29 -20.478 <.0001
##  11 - 12    -5.07 0.247 29 -20.478 <.0001
##
## P value adjustment: tukey method for comparing a family of 3 estimates
```

In this very simple case, the slope parameter is easily available as a parameter value, but we could use the `emtrends()` function to obtain the slope.

```r
emtrends( model, ~Girth, 'Girth' )
```

```
##  Girth Girth.trend    SE df lower.CL upper.CL
##   13.2        5.07 0.247 29     4.56     5.57
##
## Confidence level used: 0.95
```

This output is a bit mysterious because because of the `13.248` component. What has happened is that `emtrends` is telling us the slope of the line at a particular point on the x-axis (the mean of all the girth values). While this doesn't matter in this example, because the slope is the same for all values of girth, if we had fit a quadratic model, it would not.

```r
model <- lm( Volume ~ poly(Girth, 2), data=trees )      # Girth + Girth^2
trees <- trees %>%
  dplyr::select(Volume, Girth) %>%
  cbind(predict(model, interval='conf'))

ggplot(trees, aes(x=Girth, y=Volume)) +
  geom_point() +
```

```
geom_ribbon( aes(ymin=lwr, ymax=upr), alpha=.3 ) +
geom_line( aes(y=fit) )
```
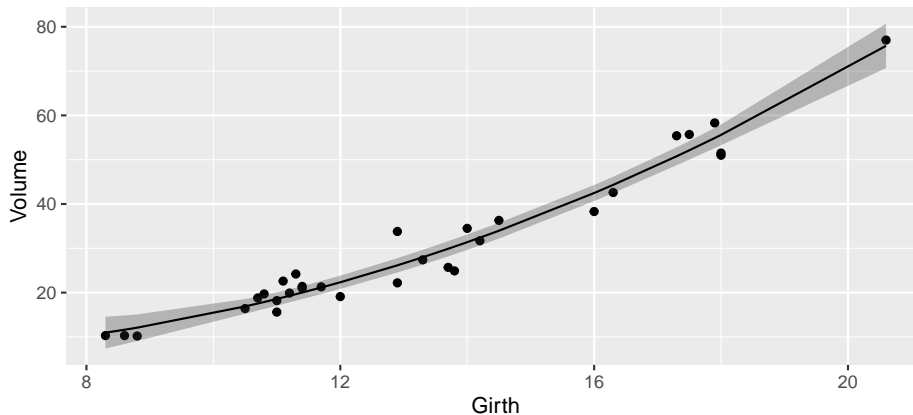


```
emtrends( model, ~ poly(Girth,2), 'Girth',
          at=list(Girth=c(10,11,12)) )
```

```
##  Girth Girth.trend    SE df lower.CL upper.CL
##     10        3.00 0.510 28     1.96     4.05
##     11        3.51 0.405 28     2.68     4.34
##     12        4.02 0.308 28     3.39     4.65
##
## Confidence level used: 0.95
```

4.4.2 1-way ANOVA

To consider the pairwise contrasts between different levels we will consider the college student hostility data again. A clinical psychologist wished to compare three methods for reducing hostility levels in university students, and used a certain test (HLT) to measure the degree of hostility. A high score on the test indicated great hostility. The psychologist used 24 students who obtained high and nearly equal scores in the experiment. Eight subjects were selected at random from among the 24 problem cases and were treated with method 1, seven of the remaining 16 students were selected at random and treated with method 2 while the remaining nine students were treated with method 3. All treatments were continued for a one-semester period. Each student was given the HLT test at the end of the semester, with the results show in the following table.

```r
Hostility <- data.frame(
  HLT = c(96,79,91,85,83,91,82,87,
          77,76,74,73,78,71,80,
          66,73,69,66,77,73,71,70,74),
  Method = c( rep('M1',8), rep('M2',7), rep('M3',9) ) )
```

```r
ggplot(Hostility, aes(x=Method, y=HLT)) +
  geom_boxplot()
```


To use the `emmeans()`, we again will use the pairwise command where we specify that we want all the pairwise contrasts between Method levels.

```r
model <- lm( HLT ~ Method, data=Hostility )
emmeans(model, specs = pairwise~Method,
        infer=c(TRUE,TRUE) )    # Print out Confidence Intervals and Pvalues
```

```
## $emmeans
##  Method emmean   SE df lower.CL upper.CL t.ratio p.value
##  M1       86.8 1.52 21     83.6     89.9 57.141  <.0001
##  M2       75.6 1.62 21     72.2     78.9 46.563  <.0001
##  M3       71.0 1.43 21     68.0     74.0 49.604  <.0001
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate   SE df lower.CL upper.CL t.ratio p.value
##  M1 - M2     11.18 2.22 21    5.577     16.8 5.030   0.0002
##  M1 - M3     15.75 2.09 21   10.491     21.0 7.548   <.0001
##  M2 - M3      4.57 2.16 21   -0.883     10.0 2.112   0.1114
##
## Confidence level used: 0.95
```

```
## Conf-level adjustment: tukey method for comparing a family of 3 estimates
## P value adjustment: tukey method for comparing a family of 3 estimates
```

By default, `emmeans()` and `emtrends()` don't provide p-values for testing if the effect could possibly be equal to zero. To get those, you'll need to set the inference option to return both confidence intervals and p-values using the `infer=c(TRUE,TRUE)`, where the two logical values control the inclusion of the confidence interval and p-value, respectively.

4.5 Exercises

1. The American Community Survey is on ongoing survey being conducted monthly by the US Census Bureau and the package `Lock5Data` has a dataset called `EmployedACS` that has 431 randomly selected anonymous US residents from this survey.

```
data('EmployedACS', package='Lock5Data')
?Lock5Data::EmployedACS
```

   a) Create a boxplot of the respondents' `Race` and `Income`.
   b) Fit a 1-way ANOVA on these data.
   c) Use `multcomp::glht()` to calculate all the pairwise differences between the race categories.
   d) Use `emmeans()` to calculate all the pairwise differences between the race categories.

2. We will examine a data set from Ashton et al. (2007) that relates the length of a tortoise's carapace to the number of eggs laid in a clutch. The data are

```
Eggs <- data.frame(
  carapace    = c(284,290,290,290,298,299,302,306,306,
                  309,310,311,317,317,320,323,334,334),
  clutch.size = c(3,2,7,7,11,12,10,8,8,
                  9,10,13,7,9,6,13,2,8))
```

   a) Plot the data with carapace as the explanatory variable and clutch size as the response.
   b) We want to fit the model

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i \text{ where } \epsilon_i \stackrel{iid}{\sim} N(0, \sigma^2)$$

   To fit this model, we need the design matrix with a column of ones, a column of x_i values and a column of x_i^2 values. To fit this model, we could create a new column of x_i^2 values, or do it in the formula.

```
Eggs <- Eggs %>% mutate( carapace.2 = carapace^2 )

model <- lm(clutch.size ~ 1 + carapace + carapace.2, data=Eggs)    # one version
model <- lm(clutch.size ~ 1 + carapace + I(carapace^2), data=Eggs) # Do squaring inside
model <- lm(clutch.size ~ 1 + poly(carapace, degree=2, raw=TRUE), data=Eggs) # using pol
```

c) Use the `predict()` function to calculate the regression predictions and add those predictions to the `Eggs` dataset.

d) Graph the data with the regression curve.

e) Use the `emmeans()` function to estimate the clutch sizes for tortoise's with carapace sizes of 300 and 320. Provide a confidence interval for the estimates and a p-value for the hypothesis test that the value could be zero.

f) Use the `emmeans()` function to estimate the difference between clutch size prediction for tortoise's with carapace sizes of 300 and 320. Provide a confidence interval for the estimates and a p-value for the hypothesis test that the difference could be zero.

g) Use the `emtrends()` function to find the estimated slope at carapace sizes of 300 and 320. Provide a confidence interval for the estimates and a p-value for the hypothesis test that the value could be zero.

h) Use the `emtrends()` function to estimate the differences between slopes at carapace sizes 300 and 320. Provide a confidence interval for the estimates and a p-value for the hypothesis test that the difference could be zero.

Statistical Models

Chapter 5

Analysis of Covariance (ANCOVA)

```
library(tidyverse)   # ggplot2, tidyr, dplyr
library(emmeans)
```

5.1 Introduction

One way that we could extend the ANOVA and regression models is to have both categorical and continuous predictor variables. For historical reasons going back to pre "computer in your pocket" days, statisticians call this the *Analysis of Covariance* (ANCOVA) model. Because it is just another example of a $y = X\beta + \epsilon$ linear model, I prefer to think of it as simply having both continuous and categorical variables in my model. None of the cookbook calculations change, but the interpretation of the parameters gets much more interesting.

The dataset `teengamb` in the package `faraway` has data regarding the rates of gambling among teenagers in Britain and their gender and socioeconomic status. One question we might be interested in is how gender and income relate to how much a person gambles. But what should be the effect of gender be?

There are two possible ways that gender could enter the model. Either:

1. We could fit two lines to the data one for males and one for females but require that the lines be parallel (i.e. having the same slopes for income). This is accomplished by having a separate y-intercept for each gender. In effect, the line for the females would be offset by a constant amount from the male line.

2. We could fit two lines but but allow the slopes to differ as well as the y-intercept. This is referred to as an "interaction" between income and gender. One way to remember that this is an interaction is because the effect of income on gambling rate is dependent on the gender of the individual.
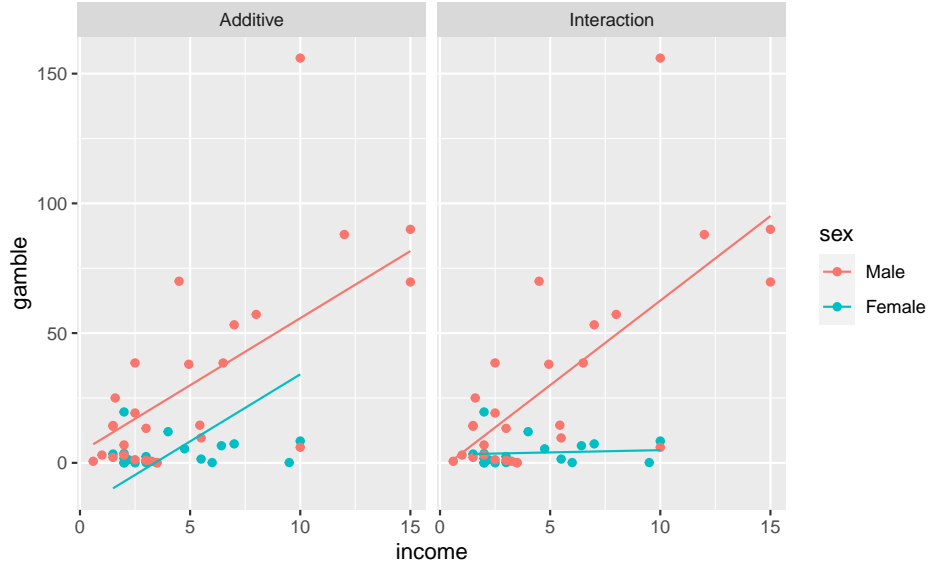
*It should be noted here, that the constant variance assumption is being violated and we really ought to do a transformation. I would recommend performing a $\sqrt{\cdot}$ transformation on both the `gamble` and `income` covariates, but we'll leave them as is for now.

We will now see how to go about fitting these two models. As might be imagined, these can be fit in the same fashion we have been solving the linear models, but require a little finesse in defining the appropriate design matrix X.

5.2   Offset parallel Lines (aka additive models)

In order to get offset parallel lines, we want to write a model

$$y_i = \begin{cases} \beta_0 + \beta_1 + \beta_2 x_i + \epsilon_i & \text{if female} \\ \beta_0 + \beta_2 x_i + \epsilon_i & \text{if male} \end{cases}$$

where β_1 is the vertical offset of the female group regression line to the reference group, which is the males regression line. Because the first 19 observations are

female, we can this in in matrix form as

$$
\begin{bmatrix} y_1 \\ \vdots \\ y_{19} \\ y_{20} \\ \vdots \\ y_{47} \end{bmatrix}
=
\begin{bmatrix} 1 & 1 & x_1 \\ \vdots & \vdots & \vdots \\ 1 & 1 & x_{19} \\ 1 & 0 & x_{20} \\ \vdots & \vdots & \vdots \\ 1 & 0 & x_{47} \end{bmatrix}
\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}
+
\begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_{19} \\ \epsilon_{20} \\ \vdots \\ \epsilon_{47} \end{bmatrix}
$$

I like this representation where β_1 is the offset from the male regression line because it makes it very convenient to test if the offset is equal to zero. The second column of the design matrix referred to as a "dummy variable" or "indicator variable" that codes for the female gender. Notice that even though I have two genders, I only had to add one additional variable to my model because we already had a y-intercept β_0 and we only added one indicator variable for females.

What if we had a third group? Then we would fit another column of indicator variable for the third group. The new beta coefficient in the model would be the offset of the new group to the reference group. For example we consider $n = 9$ observations with $n_i = 3$ observations per group where $y_{i,j}$ is the j th replication of the i th group.

$$
\begin{bmatrix} y_{1,1} \\ y_{1,2} \\ y_{1,3} \\ y_{2,1} \\ y_{2,2} \\ y_{2,3} \\ y_{3,1} \\ y_{3,2} \\ y_{3,3} \end{bmatrix}
=
\begin{bmatrix} 1 & 0 & 0 & x_{1,1} \\ 1 & 0 & 0 & x_{1,2} \\ 1 & 0 & 0 & x_{1,3} \\ 1 & 1 & 0 & x_{2,1} \\ 1 & 1 & 0 & x_{2,2} \\ 1 & 1 & 0 & x_{2,3} \\ 1 & 0 & 1 & x_{3,1} \\ 1 & 0 & 1 & x_{3,2} \\ 1 & 0 & 1 & x_{3,3} \end{bmatrix}
\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}
+
\begin{bmatrix} \epsilon_{1,1} \\ \epsilon_{1,2} \\ \epsilon_{1,3} \\ \epsilon_{2,1} \\ \epsilon_{2,2} \\ \epsilon_{2,3} \\ \epsilon_{3,3} \\ \epsilon_{3,2} \\ \epsilon_{3,3} \end{bmatrix}
$$

In this model, β_0 is the y-intercept for group 1. The parameter β_1 is the vertical offset from the reference group (group 1) for the second group. Similarly β_2 is the vertical offset for group 3. All groups will share the same slope, β_4.

5.3 Lines with different slopes (aka Interaction model)

We can now include a discrete random variable and create regression lines that are parallel, but often that is inappropriate, such as in the teenage gambling dataset. We want to be able to fit a model that has different slopes.

$$
y_i = \begin{cases} (\beta_0 + \beta_1) + (\beta_2 + \beta_3)\, x_i + \epsilon_i & \text{if female} \\ \beta_0 + \beta_2 x_i + \epsilon_i & \text{if male} \end{cases}
$$

Where β_1 is the offset in y-intercept of the female group from the male group, and β_3 is the offset in slope. Now our matrix formula looks like

$$
\begin{bmatrix} y_1 \\ \vdots \\ y_{19} \\ y_{20} \\ \vdots \\ y_{47} \end{bmatrix}
=
\begin{bmatrix}
1 & 1 & x_1 & x_1 \\
\vdots & \vdots & \vdots & \vdots \\
1 & 1 & x_{19} & x_{19} \\
1 & 0 & x_{20} & 0 \\
\vdots & \vdots & \vdots & \vdots \\
1 & 0 & x_{47} & 0
\end{bmatrix}
\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}
+
\begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_{19} \\ \epsilon_{20} \\ \vdots \\ \epsilon_{47} \end{bmatrix}
$$

where the new fourth column is the what I would get if I multiplied the x column element-wise with the dummy-variable column. To fit this model in R we have

```r
data('teengamb', package='faraway')

# Forces R to recognize that 0, 1 are categorical, also
# relabels the levels to something I understand.
teengamb <- teengamb %>% mutate( sex = ifelse( sex==1, 'Female', 'Male') )

# Fit a linear model with the interaction of sex and income
# Interactions can be specified useing a colon :
m1 <- lm( gamble ~ 1 + sex + income + sex:income, data=teengamb )
m1 <- lm( gamble ~     sex + income + sex:income, data=teengamb )

# R allows a shortcut for the prior definition
m1 <- lm( gamble ~ sex * income, data=teengamb )

# save the fit, lwr, upr values for each observation
# these are the yhat and CI
# If columns for fit, upr, lwr are already present, remove them
teengamb <- teengamb %>%
  dplyr::select( -matches('fit'), -matches('lwr'), -matches('upr') ) %>%
  cbind( predict(m1, interval='conf') )

# Make a nice plot that includes the regression line.
ggplot(teengamb, aes(x=income, col=sex, fill=sex)) +
    geom_ribbon(aes(ymin=lwr, ymax=upr),
                  alpha=.3) +    # how solid the layer is
    geom_point(aes(y=gamble)) +
    geom_line(aes(y=fit))
```

```
# print the model summary
summary(m1)
```

```
##
## Call:
## lm(formula = gamble ~ sex * income, data = teengamb)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -56.522  -4.860  -1.790   6.273  93.478
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.1400     9.2492   0.339  0.73590
## sexMale        -5.7996    11.2003  -0.518  0.60724
## income          0.1749     1.9034   0.092  0.92721
## sexMale:income  6.3432     2.1446   2.958  0.00502 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.98 on 43 degrees of freedom
## Multiple R-squared:  0.5857, Adjusted R-squared:  0.5568
## F-statistic: 20.26 on 3 and 43 DF,  p-value: 2.451e-08
```

To interpret the terms, we have

Coefficients	Interpretation
(Intercept)	y-intercept for the females
sexMale	The difference in y-intercept for Males
income	Slope of the female regression line
sexMale:income	The offset in slopes for the Males

So looking at the summary, we see the interaction term `sexMale:income` is statistically significant indicating that we prefer the more complicated model with different slopes for each gender.

To calculate the differences between the predicted values at an `income` levels of 5 and 10, we could use `multcomp::glht()` and figure out the appropriate contrast vector, but we'll use the easy version with `emmeans()`

```
emmeans(m1, specs = ~ income * sex,
        at=list(income=c(5,10), sex=c('Male','Female')))
```

```
##  income sex     emmean    SE df lower.CL upper.CL
##       5 Male     29.93  3.97 43    21.93     37.9
##      10 Male     62.52  6.35 43    49.71     75.3
##       5 Female    4.01  5.08 43    -6.23     14.3
##      10 Female    4.89 12.13 43   -19.58     29.4
##
## Confidence level used: 0.95
```

If we are interested in the differences we can just do a `pairwise` in the `specs` argument, but I also want to just calculate the differences at each income level.

```
# The pipe in the formula is essentially a group_by
emmeans(m1, specs = pairwise ~ sex | income,
        at=list(income=c(5,10),
                sex=c('Male','Female')))
```

```
## $emmeans
## income =  5:
##  sex     emmean    SE df lower.CL upper.CL
##  Male     29.93  3.97 43    21.93     37.9
##  Female    4.01  5.08 43    -6.23     14.3
##
## income = 10:
##  sex     emmean    SE df lower.CL upper.CL
##  Male     62.52  6.35 43    49.71     75.3
##  Female    4.89 12.13 43   -19.58     29.4
##
## Confidence level used: 0.95
##
## $contrasts
## income =  5:
##  contrast      estimate    SE df t.ratio p.value
##  Male - Female     25.9  6.44 43   4.022  0.0002
##
```

```
## income = 10:
##  contrast        estimate    SE df t.ratio p.value
##  Male - Female       57.6 13.69 43 4.208   0.0001
```

If we want the slopes as well as the difference in slopes, we would use the `emtrends()` function.

```r
emtrends(m1, pairwise ~ income * sex, var = "income",
        at=list(income=10, sex=c('Male','Female')))
```

```
## $emtrends
##  income sex     income.trend    SE df lower.CL upper.CL
##      10 Male           6.518 0.988 43     4.53     8.51
##      10 Female         0.175 1.903 43    -3.66     4.01
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast             estimate    SE df t.ratio p.value
##  10 Male - 10 Female      6.34 2.14 43 2.958   0.0050
```

While I specified to calculate the slope at the x-value of `income=10`, that doesn't matter because the slopes are the same at all x-values

Somewhat less interestingly, we could calculate the average of the Male and Female slopes.

```r
# when specs doesn't include a variable that was used
# in the model, this will either
#   a) average over the missing levels (categorical)
#   b) use the average value of the variable (quantitative)
emtrends(m1, specs = ~ income, var = 'income',
        at=list(income=10))
```

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

```
##  income income.trend    SE df lower.CL upper.CL
##      10         3.35 1.07 43     1.18     5.51
##
## Results are averaged over the levels of: sex
## Confidence level used: 0.95
```

5.4   Iris Example

For a second example, we will explore the relationship between sepal length and sepal width for three species of irises. This data set is available in R as `iris`.

```r
data(iris)            # read in the iris dataset
levels(iris$Species)  # notice the order of levels of Species
```

```
## [1] "setosa"     "versicolor" "virginica"
```

The very first thing we should do when encountering a dataset is to do some sort of graphical summary to get an idea of what model seems appropriate.

```r
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) +
  geom_point()
```


Looking at this graph, it seems that I will likely have a model with different y-intercepts for each species, but it isn't clear to me if we need different slopes.

We consider the sequence of building successively more complex models:

```r
# make virginica the reference group
iris <- iris %>%
  mutate( Species = forcats::fct_relevel(Species, 'virginica') )

m1 <- lm( Sepal.Width ~ Sepal.Length, data=iris )            # One line
m2 <- lm( Sepal.Width ~ Sepal.Length + Species, data=iris )  # Parallel Lines
m3 <- lm( Sepal.Width ~ Sepal.Length * Species, data=iris )  # Non-parallel Lines
```

The three models we consider are the following:

Looking at these, it seems obvious that the simplest model where we ignore Species is horrible. The other two models seem decent, and I am not sure about the parallel lines model vs the differing slopes model.

```
m1 %>% broom::tidy() %>% mutate_if( is.numeric, round, digits=3 )
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic p.value
##   <chr>            <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)       3.42     0.254      13.5   0
## 2 Sepal.Length     -0.062    0.043      -1.44  0.152
```

For the simplest model, there is so much unexplained noise that the slope variable isn't significant.

Moving onto the next most complicated model, where each species has their own y-intercept, but they share a slope, we have

```
m2 %>% broom::tidy() %>% mutate_if( is.numeric, round, digits=3 )
```

```
## # A tibble: 4 x 5
##   term              estimate std.error statistic p.value
##   <chr>                <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)          0.669     0.308      2.17   0.031
## 2 Sepal.Length         0.35      0.046      7.56   0
## 3 Speciessetosa        1.01      0.093     10.8    0
## 4 Speciesversicolor    0.024     0.065      0.37   0.712
```

The first two lines are the y-intercept and slope associated with the reference group and the last two lines are the y-intercept offsets from the reference group to *Setosa* and *Versicolor*, respectively. We have that the slope associated with increasing Sepal Length is significant and that *Setosa* has a statistically different y-intercept than the reference group *Virginica* and that *Versicolor* does not have a statistically different y-intercept than the reference group.

Finally we consider the most complicated model that includes two more slope parameters

```
m3 %>% broom::tidy() %>% mutate_if( is.numeric, round, digits=3 )
```

```
## # A tibble: 6 x 5
##   term                          estimate std.error statistic p.value
##   <chr>                            <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)                       1.45     0.405      3.57   0
## 2 Sepal.Length                      0.232    0.061      3.79   0
## 3 Speciessetosa                    -2.02     0.686     -2.94   0.004
## 4 Speciesversicolor                -0.574    0.605     -0.95   0.344
## 5 Sepal.Length:Speciessetosa        0.567    0.126      4.49   0
## 6 Sepal.Length:Speciesversicolor    0.088    0.097      0.905  0.367
```

These parameters are:

Meaning	R-label
Reference group y-intercept	(Intercept)
Reference group slope	Sepal.Length
offset to y-intercept for Setosa	Speciessetosa
offset to y-intercept for Versicolor	Speciesversicolor
offset to slope for Setosa	Sepal.Length:Speciessetosa
offset to slope for Versicolor	Sepal.Length:Speciesversicolor

It appears that slope for *Setosa* is different from the reference group *Virginica.* However because we've added 2 parameters to the model, testing Model2 vs Model3 is not equivalent to just looking at the p-value for that one slope. Instead we need to look at the F-test comparing the two models which will evaluate if the decrease in SSE is sufficient to justify the addition of two parameters.

```
anova(m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: Sepal.Width ~ Sepal.Length + Species
## Model 2: Sepal.Width ~ Sepal.Length * Species
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    146 12.193
## 2    144 10.680  2    1.5132 10.201 7.19e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The F-test concludes that there is sufficient decrease in the SSE to justify adding two additional parameters to the model.

5.5 Exercises

1. The in the **faraway** package, there is a dataset named **phbirths** that gives babies birth weights along with their gestational time in utero along with the mother's smoking status.

   a. Load and inspect the dataset using

   ```
   data('phbirths', package='faraway')   # load the data within the package
   ?faraway::phbirths                     # Look at the help file
   ```

   b. Create a plot of the birth weight vs the gestational age. Color code the points based on the mother's smoking status. Does it appear that smoking matters?

   c. Fit the simple model (one regression line) along with both the main effects (parallel lines) and interaction (non-parallel lines) ANCOVA model to these data. Which model is preferred?

   d. Using whichever model you selected in the previous section, create a graph of the data along with the confidence region for the regression line(s).

   e. Now consider only the "full term babies" which are babies with gestational age at birth ≥ 36 weeks. With this reduced dataset, repeat parts c,d.

    f. Interpret the relationship between gestational length and mother's smoking status on birth weight.

2. The in the `faraway` package, there is a dataset named `clot` that gives information about the time for blood to clot verses the blood dilution concentration when the blood was diluted with prothrombin-free plasma. Unfortunately the researchers had to order the plasma in two different lots (could think of this as two different sources) and need to ascertain if the lot number makes any difference in clotting time.

    a. Log transform the `time` and `conc` variable and plot the log-transformed data with color of the data point indicating the lot number. *(We will discuss why we performed this transformation later in the course.)*

    b. Ignoring the slight remaining curvature in the data, perform the appropriate analysis using transformed variables. Does `lot` matter?

3. In base R, there is a data set `ToothGrowth` which is data from an experiment giving Vitamin C to guinea pigs. The guinea pigs were given vitamin C doses either via orange juice or an ascorbic acid tablet. The response of interest was a measure of tooth growth where a higher growth is better.

    a. Log transform the `dose` and use that throughout this problem. Use e as the base, which R does by default when you use the `log()` function. *(We will discuss why we performed this transformation later in the course.)*

    b. Graph the data, fit appropriate ANCOVA models, and describe the relationship between the delivery method, log(dose) level, and tooth growth. Produce a graph with the data and the regression line(s) along with the confidence region for the line(s).

    c. Is there a statistically significant difference in slopes between the two delivery methods?

    d. Just using your graphs and visual inspection, at low dose levels, say $\log(dose) = -0.7$, is there a difference in delivery method? What about at high dose levels, say $\log(dose) = 0.7$?

    e. Use `emmeans()` to test if there is a statistically significant difference at low dose levels $\log(dose) = -0.7$. Furthermore, test if there is a statistically significant difference at high dose levels. Summarize your findings.

Chapter 6

Two-way ANOVA

```
# Load my usual packages
library(tidyverse)   # ggplot2, dplyr, tidyr
library(ggfortify)   # autoplot() for lm objects
library(emmeans)     # pairwise contrasts stuff
```

6.1 Review of 1-way ANOVA

Given a categorical covariate (which I will call a factor) with I levels, we are interested in fitting the model

$$y_{ij} = \mu + \tau_i + \epsilon_{ij}$$

where $\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$, μ is the overall mean, and τ_i are the offset of factor level i from μ. Unfortunately this model is not identifiable because I could add a constant (say 5) to μ and subtract that same constant from each of the τ_i values and the group mean $\mu + \tau_i$ would not change. There are two easy restrictions we could make to make the model identifiable:

1. Set $\mu = 0$. In this case, τ_i represents the expected value of an observation in group level i. We call this the "cell means" representation.

2. Set $\tau_1 = 0$. Then μ represents the expected value of treatment 1, and the τ_i values will represent the offsets from group 1. The group or level that we set to be zero is then referred to as the reference group. We can call this the "offset from reference" model.

We will be interested in testing the null and alternative hypotheses

$$H_0: \quad y_{ij} = \mu + \epsilon_{ij}$$
$$H_a: \quad y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

6.1.1   An Example

We look at a dataset that comes from the study of blood coagulation times:
24 animals were randomly assigned to four different diets and the samples were
taken in a random order.   The diets are denoted as A,B,C,and D and the
response of interest is the amount of time it takes for the blood to coagulate.

```
data('coagulation', package='faraway')
ggplot(coagulation, aes(x=diet, y=coag)) +
    geom_boxplot() +
    labs( x='Diet', y='Coagulation Time' )
```


Just by looking at the graph, we expect to see that diets A and D are similar
while B and C are different from A and D and possibly from each other, too.
We first fit the offset model.

```
m <- lm(coag ~ diet, data=coagulation)
summary(m)
```

```
##
## Call:
## lm(formula = coag ~ diet, data = coagulation)
##
## Residuals:
##     Min      1Q Median      3Q     Max
```

```
##  -5.00  -1.25   0.00   1.25   5.00
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.100e+01  1.183e+00  51.554  < 2e-16 ***
## dietB       5.000e+00  1.528e+00   3.273 0.003803 **
## dietC       7.000e+00  1.528e+00   4.583 0.000181 ***
## dietD       2.991e-15  1.449e+00   0.000 1.000000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.366 on 20 degrees of freedom
## Multiple R-squared:  0.6706, Adjusted R-squared:  0.6212
## F-statistic: 13.57 on 3 and 20 DF,  p-value: 4.658e-05
```

Notice that diet A is the reference level and it has a mean of 61. Diet B has an offset from A of 5, etc. From the very small F-statistic, we conclude that simple model

$$y_{ij} = \mu + \epsilon_{ij}$$

is not sufficient to describe the data.

6.1.2   Degrees of Freedom

Throughout the previous example, the degrees of freedom that are reported keeps changed depending on what models we are comparing. The simple model we are considering is

$$y_{ij} \sim \mu + \epsilon_{ij}$$

which has 1 parameter that defines the expected value versus

$$y_{ij} \sim \mu + \tau_i + \epsilon_{ij}$$

where there really are only 4 parameters that define the expected value because $\tau_1 = 0$. In general, the larger model is only adding $I - 1$ terms to the model where I is the number of levels of the factor of interest.

6.1.3   Pairwise Comparisons

After detecting differences in the factor levels, we are often interested in which factor levels are different from which. Often we are interested in comparing the mean of level i with the mean of level j.

As usual we let the vector of parameter estimates be $\hat{\beta}$ then the contrast of interested can be written as

$$c^T \hat{\beta} \pm t_{n-p}^{1-\alpha/2} \, \hat{\sigma} \sqrt{c^T \left(X^T X \right)^{-1} c}$$

for some vector c.

Unfortunately this interval does not take into account the multiple comparisons issue (i.e. we are making $I(I-1)/2$ contrasts if our factor has I levels). To account for this, we will not use a quantile from a t-distribution, but from Tukey's studentized range distribution $q_{n,n-I}$ divided by $\sqrt{2}$. The intervals we will use are:

$$c^T\widehat{\beta} \pm \frac{q_{n,n-I}^{1-\alpha/2}}{\sqrt{2}}\,\widehat{\sigma}\sqrt{c^T\left(X^TX\right)^{-1}c}$$

There are several ways to make R calculate this interval, but the easiest is to use the **emmeans** package. This package computes the above intervals which are commonly known as Tukey's Honestly Significant Differences.

```r
m <- lm(coag ~ diet, data=coagulation)    # use the lm() function as usual
emmeans(m, specs= pairwise~diet) %>%
  summary(level=0.90)
```

```
## $emmeans
##  diet emmean    SE df lower.CL upper.CL
##  A        61 1.183 20     59.0     63.0
##  B        66 0.966 20     64.3     67.7
##  C        68 0.966 20     66.3     69.7
##  D        61 0.837 20     59.6     62.4
##
## Confidence level used: 0.9
##
## $contrasts
##  contrast estimate   SE df t.ratio p.value
##  A - B          -5 1.53 20  -3.273  0.0183
##  A - C          -7 1.53 20  -4.583  0.0010
##  A - D           0 1.45 20   0.000  1.0000
##  B - C          -2 1.37 20  -1.464  0.4766
##  B - D           5 1.28 20   3.912  0.0044
##  C - D           7 1.28 20   5.477  0.0001
##
## P value adjustment: tukey method for comparing a family of 4 estimates
```

Here we see that diets A and D are similar to each other, but different than B and C and that B and C are not statistically different from each other at the 0.10 level.

Often I want to produce the "Compact Letter Display" which identifies which groups are significantly different. Unfortunately this leads to a somewhat binary decision of "is statistically significant" or "is NOT statistically significant", but we should at least know how to do this calculation.

```
LetterData <- emmeans(m, specs= ~ diet) %>%  # cld() will freak out if you have pairwise here...
  multcomp::cld(Letters=letters, level=0.95) %>%
  mutate(.group = str_remove_all(.group, '\\s') ) %>%  # remove the spaces
  mutate( y = 73 )    # height to place the letters at.
LetterData
```

```
##  diet emmean    SE df lower.CL upper.CL .group  y
##  A         61 1.183 20     58.5     63.5 a      73
##  D         61 0.837 20     59.3     62.7 a      73
##  B         66 0.966 20     64.0     68.0 b      73
##  C         68 0.966 20     66.0     70.0 b      73
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
```

I can easily add these to my boxplot with the following:

```
ggplot(coagulation, aes(x=diet, y=coag)) +
  geom_boxplot() +
  labs( x='Diet', y='Coagulation Time' ) +
  geom_text( data=LetterData, aes(x=diet, y=y, label=.group), size=8 )
```


6.2 Two-Way ANOVA

Given a response that is predicted by two different categorical variables. Suppose we denote the levels of the first factor as α_i and has I levels. The second factor has levels β_j and has J levels. As usual we let $\epsilon_{ijk} \overset{iid}{\sim} N\left(0, \sigma^2\right)$, and we wish to fit the model

$$y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}$$

which has the main effects of each covariate or possibly the model with the interaction

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

To consider what an interaction term might mean consider the role of temperature and humidity on the amount of fungal growth. You might expect to see data similar to this (where the numbers represent some sort of measure of fungal growth):

		5%	30%	60%	90%
	2C	2	4	8	16
Temperature	**10C**	3	9	27	81
	30C	4	16	64	256

In this case we see that increased humidity increases the amount of fungal growth, but the amount of increase depends on the temperature. At 2 C, the increase is humidity increases are significant, but at 10 C the increases are larger, and at 30 C the increases are larger yet. The effect of changing from one humidity level to the next *depends on which temperature level we are at*. This change in effect of humidity is an interaction effect. A memorable example is that chocolate by itself is good. Strawberries by themselves are also good. But the combination of chocolate and strawberries is a delight greater than the sum of the individual treats.

We can look at a graph of the Humidity and Temperature vs the Response and see the effect of increasing humidity changes based on the temperature level. Just as in the ANCOVA model, the interaction manifested itself in non-parallel slopes, the interaction manifests itself in non-parallel slopes when I connect the dots across the factor levels.
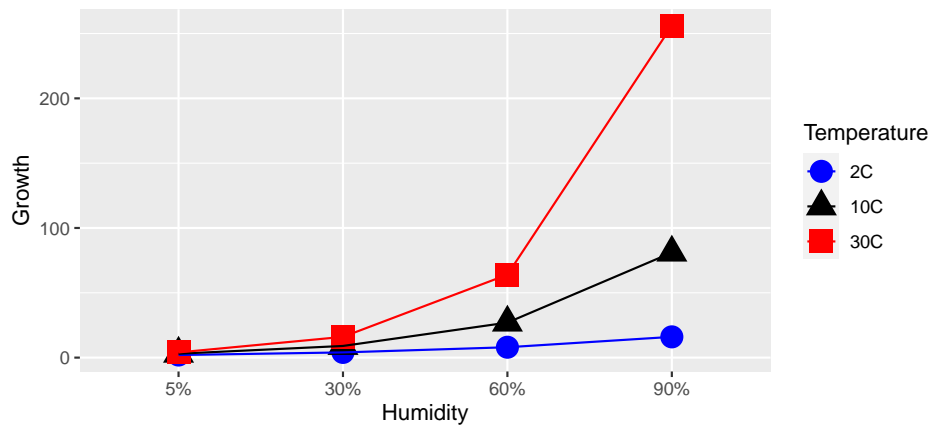
Unfortunately the presence of a significant interaction term in the model makes interpretation difficult, but examining the interaction plots can be quite helpful in understanding the effects. Notice in this example, we 3 levels of temperature and 4 levels of humidity for a total of 12 different possible treatment combinations. In general I will refer to these combinations as cells.

6.3 Orthogonality

When designing an experiment, I want to make sure than none of my covariates are confounded with each other and I'd also like for them to not be correlated. Consider the following three experimental designs, where the number in each bin is the number of subjects of that type. I am interested in testing 2 different drugs and studying its effect on heart disease within the gender groups.

Design 1	Males	Females
Treatment A	0	10
Treatment B	6	0

Design 2	Males	Females
Treatment A	1	9
Treatment B	5	1

Design 3	Males	Females
Treatment A	3	5
Treatment B	3	5

Design 4	Males	Females
Treatment A	4	4
Treatment B	4	4

1. This design is very bad. Because we have no males taking drug 1, and no females taking drug 2, we can't say if any observed differences are due to

the effect of drug 1 versus 2, or gender. When this situation happens, we say that the gender effect is confounded with the drug effect.

2. This design is not much better. Because we only have one observation in the Male-Drug 1 group, any inference we make about the effect of drug 1 on males is based on one observation. In general that is a bad idea.

3. Design 3 is better than the previous 2 because it evenly distributes the males and females among the two drug categories. However, it seems wasteful to have more females than males because estimating average of the male groups, I only have 6 observations while I have 10 females.

4. This is the ideal design, with equal numbers of observations in each gender-drug group.

Designs 3 and 4 are good because the correlation among my predictors is 0. In design 1, the drug covariate is perfectly correlated to the gender covariate. The correlation is less in design 2, but is zero in designs 3 and 4.We could show this by calculating the design matrix for each design and calculating the correlation coefficients between each of pairs of columns.

Having an orthogonal design with equal numbers of observations in each group has many nice ramifications. Most importantly, with an orthogonal design, the interpretation of parameter is not dependent on what other factors are in the model. Balanced designs are also usually optimal in the sense that the variances of $\hat{\beta}$ are as small as possible given the number of observations we have (barring any other *a priori* information).

6.4   Main Effects Model

In the one factor ANOVA case, the additional degrees of freedom used by adding a factor with I levels was $I - 1$. In the case that we consider two factors with the first factor having I levels and the second factor having J levels, then model

$$y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}$$

adds $(I - 1) + (J - 1)$ parameters to the model because both $\alpha_1 = \beta_1 = 0$.

- The intercept term, μ is the reference point for all the other parameters. This is the expected value for an observation in the first level of factor 1 and the first level of factor two.

- α_i is the amount you expect the response to increase when changing from factor 1 level 1, to factor 1 level i (while the second factor is held constant).

- β_j is the amount you expect the response to increase when changing from factor 2 level 1 to factor 2 level j (while the first factor is held constant).

Referring back to the fungus example, let the α_i values be associated with changes in humidity and β_j values be associated with changes in temperature levels. Then the expected value of each treatment combination is

	5%	30%	60%	90%
2C	$\mu + 0 + 0$	$\mu + \alpha_2 + 0$	$\mu + \alpha_3 + 0$	$\mu + \alpha_4 + 0$
10C	$\mu + 0 + \beta_2$	$\mu + \alpha_2 + \beta_2$	$\mu + \alpha_3 + \beta_2$	$\mu + \alpha_4 + \beta_2$
30C	$\mu + 0 + \beta_3$	$\mu + \alpha_2 + \beta_3$	$\mu + \alpha_3 + \beta_3$	$\mu + \alpha_4 + \beta_3$

### 6.4.1 Example - Fruit Trees

An experiment was conducted to determine the effects of four different pesticides on the yield of fruit from three different varieties of a citrus tree. Eight trees of each variety were randomly selected from an orchard. The four pesticides were randomly assigned to two trees of each variety and applications were made according to recommended levels. Yields of fruit (in bushels) were obtained after the test period.

Critically notice that we have equal number of observations for each treatment combination.

```r
# Typing the data in by hand because I got this example from a really old text book...
Pesticide <- factor(c('A','B','C','D'))
Variety <- factor(c('1','2','3'))
fruit <- data.frame( expand.grid(rep=1:2, Pest=Pesticide, Var=Variety) )
fruit$Yield <- c(49,39,50,55,43,38,53,48,55,41,67,58,53,42,85,73,66,68,85,92,69,62,85,99)
```

The first thing to do (as always) is to look at our data

```r
ggplot(fruit, aes(x=Pest, color=Var, y=Yield, shape=Var)) +
    geom_point(size=5)
```
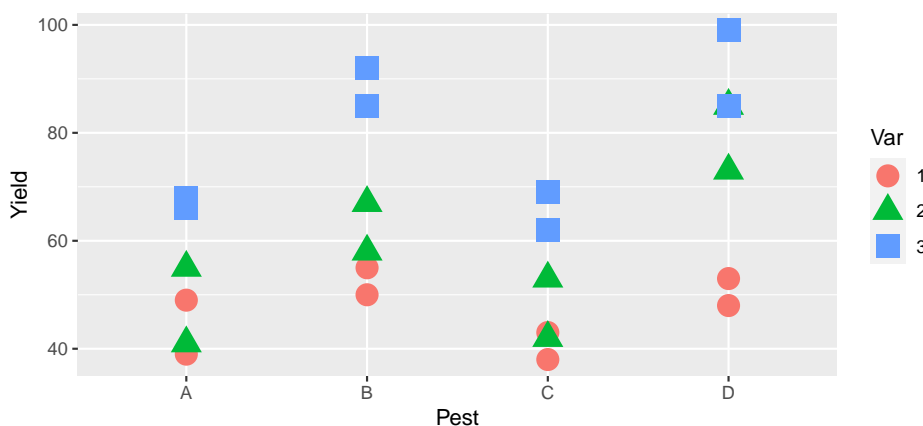
The first thing we notice is that pesticides B and D seem to be better than the others and that variety 3 seems to be the best producer. The effect of pesticide treatment seems consistent between varieties, so we don't expect that the interaction effect will be significant.

```r
m1 <- lm(Yield ~ Var, data=fruit)
m2 <- lm(Yield ~ Pest, data=fruit)
m3 <- lm(Yield ~ Var + Pest, data=fruit)
summary(m1)$coef  %>% round(digits=3)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   46.875      4.359  10.754    0.000
## Var2          12.375      6.164   2.008    0.058
## Var3          31.375      6.164   5.090    0.000
```

```r
summary(m2)$coef  %>% round(digits=3)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   53.000      6.429   8.243    0.000
## PestB         14.833      9.093   1.631    0.118
## PestC         -1.833      9.093  -0.202    0.842
## PestD         20.833      9.093   2.291    0.033
```

```r
summary(m3)$coef  %>% round(digits=3)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   38.417      3.660  10.497    0.000
## Var2          12.375      3.660   3.381    0.003
## Var3          31.375      3.660   8.573    0.000
## PestB         14.833      4.226   3.510    0.003
## PestC         -1.833      4.226  -0.434    0.670
## PestD         20.833      4.226   4.930    0.000
```

Notice that the affects for Variety and Pesticide are the same *whether or not the other is in the model*. This is due to the orthogonal design of the experiment and makes it much easier to interpret the main effects of Variety and Pesticide.

6.4.2 ANOVA Table

Most statistical software will produce an analysis of variance table when fitting a two-way ANOVA. This table is very similar to the analysis of variance table we have seen in the one-way ANOVA, but has several rows which correspond to the additional factors added to the model.

Consider the two-way ANOVA with factors A and B which have levels I and J discrete levels respectively. For convenience let RSS_1 is the residual sum of squares of the intercept-only model, and RSS_A be the residual sum of squares for the model with just the main effect of factor A, and RSS_{A+B} be the residual sum of squares of the model with both main effects. Finally assume that we have a total of n observations. The ANOVA table for this model is as follows:

Source	df	Sum of Sq (SS)	Mean Sq	F	p-value
A	$df_A = I-1$	$SS_A = RSS_1 - RSS_A$	$MS_A = SS_A/df_A$	MS_A/MSE	$P\left(F_{df_A,df_e} > F_A\right)$
B	$df_B = J-1$	$SS_B = RSS_A - RSS_{A+B}$	$MS_B = SS_B/df_B$	MS_B/MSE	$P\left(F_{df_B,df_e} > F_B\right)$
Error	$df_e = n - I - J + 1$	RSS_{A+B}	$MSE = RSS_{A+B}/df_e$		

Note, if the table is cut off, you can change decrease your font size and have it all show up...

This arrangement of the ANOVA table is referred to as "Type I" sum of squares.

We can examine this table in the fruit trees example using the `anova()()` command but just passing a single model.

```
m4 <- lm(Yield ~ Var + Pest, data=fruit)
anova( m4 )
```

```
## Analysis of Variance Table
##
## Response: Yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Var        2 3996.1 1998.04  37.292 3.969e-07 ***
## Pest       3 2227.5  742.49  13.858 6.310e-05 ***
## Residuals 18  964.4   53.58
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We might think that this is the same as fitting three nested models and running an F-test on each successive pairs of models, but it isn't. While both will give the same Sums of Squares, the F statistics are different because the MSE of the complex model is different. In particular, the F-statistics are larger and thus the p-values are smaller for detecting significant effects.

```
m1 <- lm(Yield ~ 1, data=fruit)
m2 <- lm(Yield ~ Var, data=fruit)
m3 <- lm(Yield ~ Var + Pest, data=fruit)
anova( m1, m2 )  # Notice the F-statistic here is different than previous
```

```
## Analysis of Variance Table
##
## Model 1: Yield ~ 1
## Model 2: Yield ~ Var
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     23 7188.0
## 2     21 3191.9  2    3996.1 13.146 0.0001987 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova( m2, m3 )  # This F-statistic matches what we saw previously
```

```
## Analysis of Variance Table
##
## Model 1: Yield ~ Var
## Model 2: Yield ~ Var + Pest
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1     21 3191.9
## 2     18  964.4  3    2227.5 13.858 6.31e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.4.3   Estimating Contrasts

As in the one-way ANOVA, we are interested in which factor levels differ. For example, we might suspect that it makes sense to group pesticides B and D together and claim that they are better than the group of A and C.

Just as we did in the one-way ANOVA model, this is such a common thing to do that there is an easy way to do this, using emmeans.

```
m3 <- lm(Yield ~ Var + Pest, data=fruit)
emmeans(m3, spec=pairwise~Var)
```

```
## $emmeans
##  Var emmean   SE df lower.CL upper.CL
##  1     46.9 2.59 18     41.4     52.3
##  2     59.2 2.59 18     53.8     64.7
```

```
##  3      78.2 2.59 18      72.8      83.7
##
## Results are averaged over the levels of: Pest
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  1 - 2       -12.4 3.66 18 -3.381  0.0089
##  1 - 3       -31.4 3.66 18 -8.573  <.0001
##  2 - 3       -19.0 3.66 18 -5.191  0.0002
##
## Results are averaged over the levels of: Pest
## P value adjustment: tukey method for comparing a family of 3 estimates
```

```r
emmeans(m3, spec=pairwise~Pest)
```

```
## $emmeans
##  Pest emmean    SE df lower.CL upper.CL
##  A      53.0 2.99 18     46.7     59.3
##  B      67.8 2.99 18     61.6     74.1
##  C      51.2 2.99 18     44.9     57.4
##  D      73.8 2.99 18     67.6     80.1
##
## Results are averaged over the levels of: Var
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  A - B      -14.83 4.23 18 -3.510  0.0122
##  A - C        1.83 4.23 18  0.434  0.9719
##  A - D      -20.83 4.23 18 -4.930  0.0006
##  B - C       16.67 4.23 18  3.944  0.0048
##  B - D       -6.00 4.23 18 -1.420  0.5038
##  C - D      -22.67 4.23 18 -5.364  0.0002
##
## Results are averaged over the levels of: Var
## P value adjustment: tukey method for comparing a family of 4 estimates
```

These outputs are nice and they show the main effects of variety and pesticide.
Similar to the 1-way ANOVA, we also want to be able to calculate the compact
letter display.

```r
m3 <- lm(Yield ~ Var + Pest, data=fruit)
emmeans(m3, spec= ~Var)  %>% multcomp::cld(Letters=letters)
```

```
##  Var emmean    SE df lower.CL upper.CL .group
##  1     46.9 2.59 18     41.4     52.3  a
##  2     59.2 2.59 18     53.8     64.7   b
##  3     78.2 2.59 18     72.8     83.7    c
##
## Results are averaged over the levels of: Pest
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 3 estimates
## significance level used: alpha = 0.05
```

```r
emmeans(m3, spec= ~Pest) %>% multcomp::cld(Letters=letters)
```

```
##  Pest emmean    SE df lower.CL upper.CL .group
##  C       51.2 2.99 18     44.9     57.4  a
##  A       53.0 2.99 18     46.7     59.3  a
##  B       67.8 2.99 18     61.6     74.1   b
##  D       73.8 2.99 18     67.6     80.1   b
##
## Results are averaged over the levels of: Var
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
```

So we see that each variety is significantly different from all the others and among the pesticides, A and C are indistinguishable as are B and D, but there is a difference between the A, C and B, D groups.

6.5   Interaction Model

When the model contains the interaction of the two factors, our model is written as

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

Interpreting effects effects can be very tricky. Under the interaction, the effect of changing from factor 1 level 1 to factor 1 level i depends on what level of factor 2 is. In essence, we are fitting a model that allows each of the $I \times J$ cells in my model to vary independently. As such, the model has a total of $I \times J$ parameters but because the model without interactions had $1 + (I-1) + (J-1)$ terms in it, the interaction is adding df_{AB} parameters. We can solve for this

via:

$$I \times J = 1 + (I-1) + (J-1) + df_{AB}$$
$$I \times J = I + J - 1 + df_{AB}$$
$$IJ - I - J = -1 + df_{AB}$$
$$I(J-1) - J = -1 + df_{AB}$$
$$I(J-1) - J + 1 = df_{AB}$$
$$I(J-1) - (J-1) = df_{AB}$$
$$(I-1)(J-1) = df_{AB}$$

This makes sense because the first factor added $(I-1)$ columns to the design matrix and an interaction with a continuous covariate just multiplied the columns of the factor by the single column of the continuous covariate. Creating an interaction of two factors multiplies each column of the first factor by all the columns defined by the second factor.

The expected value of the ij combination is $\mu + \alpha_i + \beta_j + (\alpha\beta)_{ij}$. Returning to our fungus example, the expected means for each treatment under the model with main effects and the interaction is

	5%	30%	60%	90%
2C	$\mu + 0 + 0 + 0$	$\mu + \alpha_2 + 0 + 0$	$\mu + \alpha_3 + 0 + 0$	$\mu + \alpha_4 + 0 + 0$
10C	$\mu + 0 + \beta_2 + 0$	$\mu + \alpha_2 + \beta_2 + (\alpha\beta)_{22}$	$\mu + \alpha_3 + \beta_2 + (\alpha\beta)_{32}$	$\mu + \alpha_4 + \beta_2 + (\alpha\beta)_{42}$
30C	$\mu + 0 + \beta_3 + 0$	$\mu + \alpha_2 + \beta_3 + (\alpha\beta)_{23}$	$\mu + \alpha_3 + \beta_2 + (\alpha\beta)_{33}$	$\mu + \alpha_4 + \beta_2 + (\alpha\beta)_{43}$

Notice that we have added $6 = 3 \cdot 2 = (4-1)(3-1) = (I-1)(J-1)$ interaction parameters $(\alpha\beta)_{ij}$ to the main effects only model. The interaction model has $p = 12$ parameters, one for each cell in my treatment array.

In general it is hard to interpret the meaning of α_i, β_j, and $(\alpha\beta)_{ij}$ and the best way to make sense of them is to look at the interaction plots.

6.5.1 ANOVA Table

Most statistical software will produce an analysis of variance table when fitting a two-way ANOVA. This table is very similar to the analysis of variance table we have seen in the one-way ANOVA, but has several rows which correspond to the additional factors added to the model.

Consider the two-way ANOVA with factors A and B which have levels I and J discrete levels respectively. For convenience let RSS_1 be the residual sum of

squares of the intercept-only model, and RSS_A be the residual sum of squares for the model with just the main effect of factor A. Likewise RSS_{A+B} and RSS_{A*B} shall be the residual sum of squares of the model with just the main effects and the model with main effects and the interaction. Finally assume that we have a total of n observations. The ANOVA table for this model is as follows:

	df	Sum Sq (SS)	MS	F	$Pr(\geq F)$
A	$df_A = I-1$	$SS_A = RSS_1 - RSS_A$	$MS_A = SS_A/df_A$	MS_A/MSE	$P(F_{df_A,df_\epsilon} \geq F_A)$
B	$df_B = J-1$	$SS_B = RSS_A - RSS_{A+B}$	$MS_B = SS_B/df_B$	MS_B/MSE	$P(F_{df_B,df_\epsilon} \geq F_B)$
AB	$df_{AB} = (I-1)(J-1)$	$SS_{A*B} = RSS_{A+B} - RSS_{A*B}$	$MS_{AB} = SS_{AB}/df_{AB}$	MS_{AB}/MSE	$P(F_{df_{AB},df_\epsilon} \geq F_{AB})$
Error	$df_\epsilon = n - IJ$	RSS_{A*B}	$MSE = RSS_{A*B}/df_\epsilon$		

This arrangement of the ANOVA table is referred to as "Type I" sum of squares. Type III sums of squares are the difference between the full interaction model and the model removing each parameter group, even when it doesn't make sense. For example in the Type III table, $SS_A = RSS_{B+A:B} - RSS_{A*B}$. There is an intermediate form of the sums of squares called Type II, that when removing a main effect also removes the higher order interaction. In the case of balanced (orthogonal) designs, there is no difference between the different types, but for non-balanced designs, the numbers will change. To access these other types of sums of squares, use the `Anova()` function in the package `car`.

6.5.2 Example - Fruit Trees (continued)

We next consider whether or not to include the interaction term to the fruit tree model. We fit the model with the interaction and then graph the results.

```r
# Create the Interaction Plot using emmeans package.  IP stands for interaction plot
m4 <- lm(Yield ~ Var * Pest, data=fruit)
emmip( m4, Var ~ Pest )  # color is LHS of the formula
```
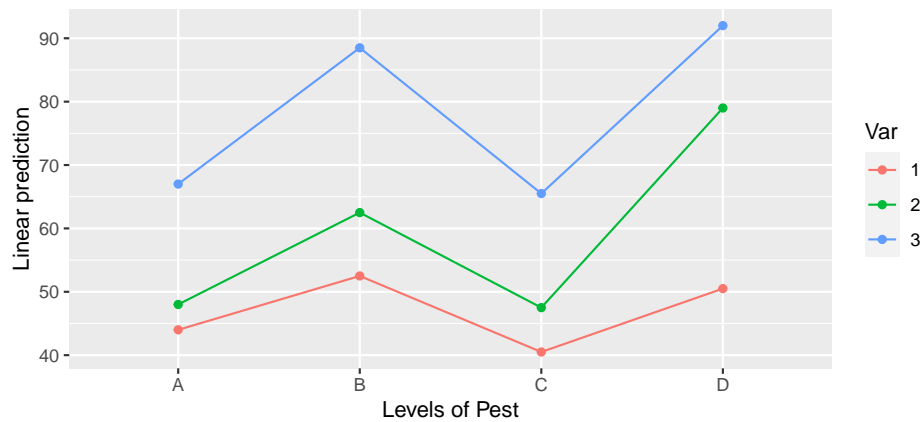
```
# Create the interaction plot by hand
m4 <- lm(Yield ~ Var * Pest, data=fruit)
fruit$y.hat <- predict(m4)
ggplot(fruit, aes(x=Pest, color=Var, shape=Var, y=Yield)) +
  geom_point(size=5) +
  geom_line(aes(y=y.hat, x=as.integer(Pest)))
```
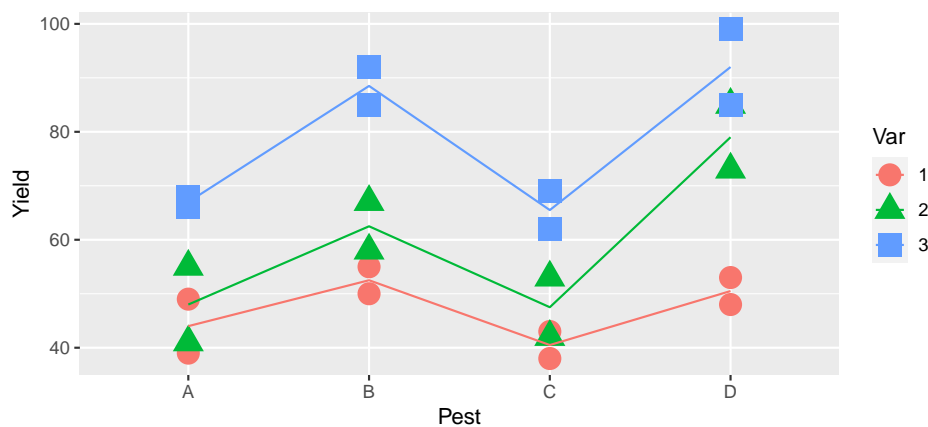

All of the line segments are close to parallel so, we don't expect the interaction
to be significant.

```
anova( m4 )
```

```
## Analysis of Variance Table
##
## Response: Yield
##            Df Sum Sq Mean Sq F value    Pr(>F)
## Var         2 3996.1 1998.04 47.2443 2.048e-06 ***
```

```
## Pest       3 2227.5  742.49 17.5563 0.0001098 ***
## Var:Pest   6  456.9   76.15  1.8007 0.1816844
## Residuals 12  507.5   42.29
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Examining the ANOVA table, we see that the interaction effect is not significant and we will stay with simpler model `Yield~Var+Pest`.

6.5.3   Example - Warpbreaks

This data set looks at the number of breaks that occur in two different types of wool under three different levels of tension (low, medium, and high). The fewer number of breaks, the better.

As always, the first thing we do is look at the data. In this case, it looks like the number of breaks decreases with increasing tension and perhaps wool B has fewer breaks than wool A.

```
data(warpbreaks)
ggplot(warpbreaks, aes(x=tension, y=breaks, color=wool, shape=wool), size=2) +
  geom_boxplot() +
  geom_point(position=position_dodge(width=.35)) # offset the wool groups
```


We next fit our linear model and examine the diagnostic plots.

```
model <- lm(breaks ~ tension + wool, data=warpbreaks)
autoplot(model, which=c(1,2)) + geom_point( aes(color=tension:wool))
```

```
## Warning: `arrange_()` is deprecated as of dplyr 0.7.0.
```

```
## Please use `arrange()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```


The residuals vs fitted values plot is a little worrisome and appears to be an issue with non-constant variance, but the normality assumption looks good. We'll check for a Box-Cox transformation next.

```
MASS::boxcox(model)
```
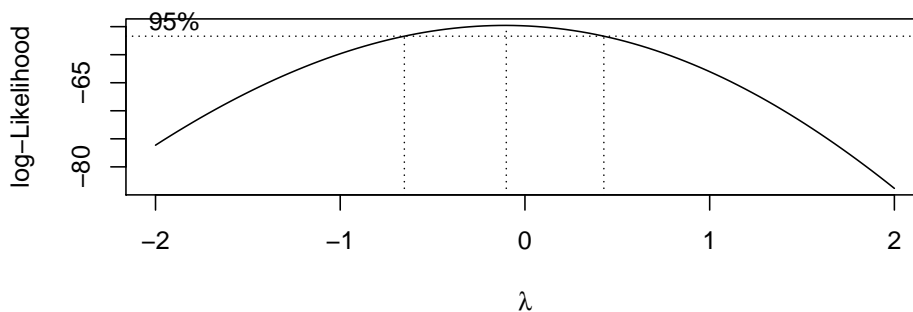

This suggests we should make a log transformation, though because the confidence interval is quite wide we might consider if the increased difficulty in interpretation makes sufficient progress towards making the data meet the model assumptions. The diagnostic plots of the resulting model look better for the constant variance assumption, but the normality is now a worse off. Because the Central Limit Theorem helps deal with the normality question, I'd rather stabilize the variance at the cost of the normality.

```
model.1 <- lm(log(breaks) ~ tension + wool, data=warpbreaks)
autoplot(model.1, which=c(1,2)) + geom_point( aes(color=tension:wool))
```

Next we'll fit the interaction model and check the diagnostic plots. The diagnostic plots look good and this appears to be a legitimate model.

```
model.2 <- lm(log(breaks) ~ tension * wool, data=warpbreaks)
autoplot(model.2, which=c(1,2)) + geom_point( aes(color=tension:wool))
```
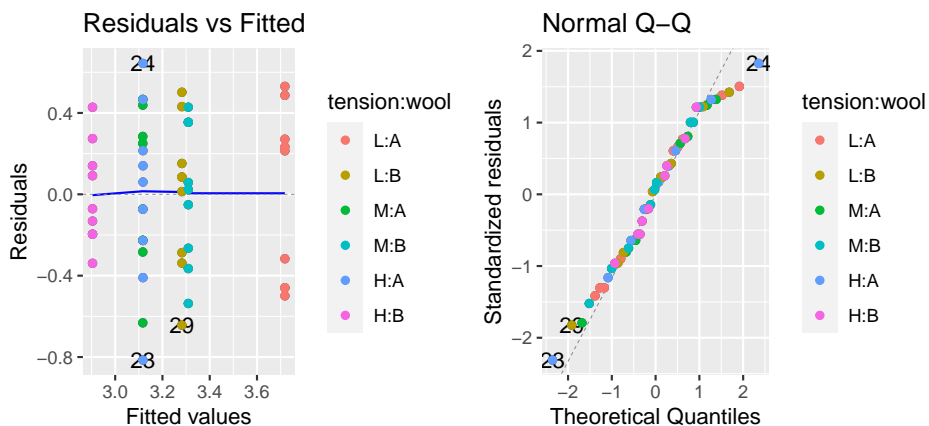

Then we'll do an F-test to see if it is a better model than the main effects model. The p-value is marginally significant, so we'll keep the interaction in the model, but recognize that it is a weak interaction.

```
anova(model.1, model.2)  # explicitly look model1 vs model2
```

```
## Analysis of Variance Table
##
## Model 1: log(breaks) ~ tension + wool
## Model 2: log(breaks) ~ tension * wool
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
```

```
## 1      50 7.6270
## 2      48 6.7138  2   0.91315 3.2642 0.04686 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(model.2)                 # table of sequentially added terms in model 2
```

```
## Analysis of Variance Table
##
## Response: log(breaks)
##              Df Sum Sq Mean Sq F value    Pr(>F)
## tension       2 2.1762 1.08808  7.7792 0.001185 **
## wool          1 0.3125 0.31253  2.2344 0.141511
## tension:wool  2 0.9131 0.45657  3.2642 0.046863 *
## Residuals    48 6.7138 0.13987
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Next we look at the effect of the interaction and the easiest way to do this is
to look at the interaction plot. The `emmeans::emmip()` just shows the mean of
each treatment combination, while the plot I made by hand shows the mean of
each treatment combination along with the raw data.

```r
A <- emmip(model.2, wool ~ tension)  # LHS is color, RHS is the x-axis variable

B <- warpbreaks %>%
  mutate( logy.hat = predict(model.2) ) %>%
  ggplot(aes(x=tension, y=log(breaks), color=wool, shape=wool)) +
  geom_point() +
  geom_line(aes(y=logy.hat, x=as.integer(tension)))  # make tension continuous to draw the lines

cowplot::plot_grid(A,B) # Plot these version of the interaction plot side-by-side.
```
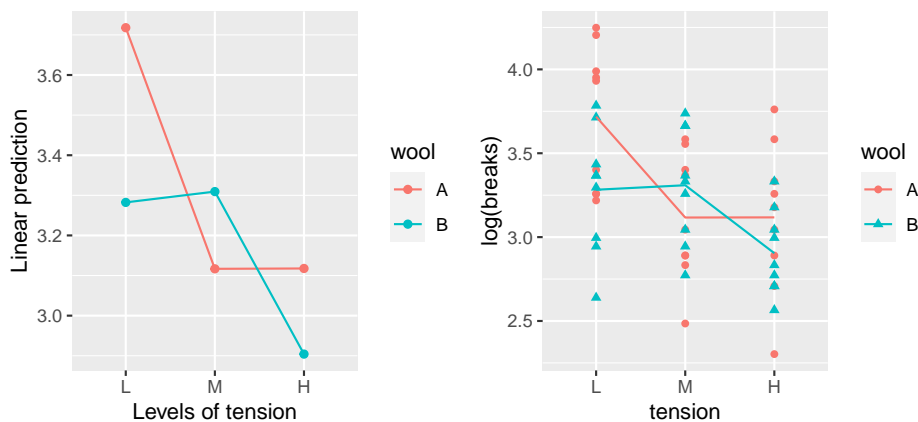
We can see that it appears that wool A has a decrease in breaks between low and medium tension, while wool B has a decrease in breaks between medium and high. It is actually quite difficult to see this interaction when we examine the model coefficients.

```
summary(model.2)
```

```
##
## Call:
## lm(formula = log(breaks) ~ tension * wool, data = warpbreaks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.81504 -0.27885  0.04042  0.27319  0.64358
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.7179     0.1247  29.824  < 2e-16 ***
## tensionM        -0.6012     0.1763  -3.410  0.00133 **
## tensionH        -0.6003     0.1763  -3.405  0.00134 **
## woolB           -0.4356     0.1763  -2.471  0.01709 *
## tensionM:woolB   0.6281     0.2493   2.519  0.01514 *
## tensionH:woolB   0.2221     0.2493   0.891  0.37749
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.374 on 48 degrees of freedom
## Multiple R-squared:  0.3363, Adjusted R-squared:  0.2672
## F-statistic: 4.864 on 5 and 48 DF,  p-value: 0.001116
```

To test if there is a statistically significant difference between medium and high tensions for wool type B, we really need to test the following hypothesis:

$$H_0 : \left(\mu + \alpha_2 + \beta_2 + (\alpha\beta)_{22}\right) - \left(\mu + \alpha_3 + \beta_2 + (\alpha\beta)_{32}\right) = 0$$
$$H_a : \left(\mu + \alpha_2 + \beta_2 + (\alpha\beta)_{22}\right) - \left(\mu + \alpha_3 + \beta_2 + (\alpha\beta)_{32}\right) \neq 0$$

This test reduces to testing if $\alpha_2 - \alpha_3 + (\alpha\beta)_{22} - (\alpha\beta)_{23} = 0$. Calculating this difference from the estimated values of the summery table we have $-.6012 + .6003 + .6281 - .2221 = 0.4051$, we don't know if that is significantly different than zero.

In the main effects model, we were able to read off the necessary test using `emmeans`. Fortunately, we can do the same thing here. In this case, we'll look at the interactions piece of the `emmeans` command. In this case, we find the test H:B - M:B in the last row of the interactions.

```
emmeans(model.2, specs= pairwise~tension*wool)
```

```
## $emmeans
##  tension wool emmean    SE df lower.CL upper.CL
##  L        A      3.72 0.125 48     3.47     3.97
##  M        A      3.12 0.125 48     2.87     3.37
##  H        A      3.12 0.125 48     2.87     3.37
##  L        B      3.28 0.125 48     3.03     3.53
##  M        B      3.31 0.125 48     3.06     3.56
##  H        B      2.90 0.125 48     2.65     3.15
##
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
##
## $contrasts
##  contrast    estimate    SE df t.ratio p.value
##  L A - M A  0.601196 0.176 48   3.410  0.0158
##  L A - H A  0.600323 0.176 48   3.405  0.0160
##  L A - L B  0.435567 0.176 48   2.471  0.1535
##  L A - M B  0.408619 0.176 48   2.318  0.2071
##  L A - H B  0.813794 0.176 48   4.616  0.0004
##  M A - H A -0.000873 0.176 48  -0.005  1.0000
##  M A - L B -0.165629 0.176 48  -0.939  0.9341
##  M A - M B -0.192577 0.176 48  -1.092  0.8821
##  M A - H B  0.212598 0.176 48   1.206  0.8319
##  H A - L B -0.164756 0.176 48  -0.935  0.9355
##  H A - M B -0.191704 0.176 48  -1.087  0.8840
##  H A - H B  0.213471 0.176 48   1.211  0.8295
##  L B - M B -0.026948 0.176 48  -0.153  1.0000
##  L B - H B  0.378227 0.176 48   2.145  0.2823
##  M B - H B  0.405175 0.176 48   2.298  0.2149
##
## Results are given on the log (not the response) scale.
## P value adjustment: tukey method for comparing a family of 6 estimates
```

The last call to **emmeans** gives us all the pairwise tests comparing the cell means.
If we don't want to wade through all the other pairwise contrasts we could do
the following:

```
# If I want to not wade through all those contrasts and just grab the
# contrasts for wool type 'B' and tensions 'M' and 'H'
emmeans(model.2, pairwise~tension*wool, at=list(wool='B', tension=c('M','H')))
```

```
## $emmeans
```

```
##  tension wool emmean    SE df lower.CL upper.CL
##  M       B      3.31 0.125 48     3.06     3.56
##  H       B      2.90 0.125 48     2.65     3.15
##
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
##
## $contrasts
##  contrast  estimate    SE df t.ratio p.value
##  M B - H B    0.405 0.176 48 2.298   0.0260
##
## Results are given on the log (not the response) scale.
```

What would happen if we just looked at the main effects? In the case where our experiment is balanced with equal numbers of observations in each treatment cell, we can interpret these differences as follows. Knowing that each cell in our table has a different estimated mean, we could consider the average of all the type A cells as the typical wool A. Likewise we could average all the cell means for the wool B cells. Then we could look at the difference between those two averages. In the balanced design, this is equivalent to removing the tension term from the model and just looking at the difference between the average log number of breaks.

```
emmeans(model.2, specs= pairwise~tension )
```

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

## $emmeans
##  tension emmean     SE df lower.CL upper.CL
##  L         3.50 0.0882 48     3.32     3.68
##  M         3.21 0.0882 48     3.04     3.39
##  H         3.01 0.0882 48     2.83     3.19
##
## Results are averaged over the levels of: wool
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  L - M       0.287 0.125 48 2.303   0.0649
##  L - H       0.489 0.125 48 3.925   0.0008
##  M - H       0.202 0.125 48 1.622   0.2465
##
## Results are averaged over the levels of: wool
## Results are given on the log (not the response) scale.
## P value adjustment: tukey method for comparing a family of 3 estimates
```

```r
emmeans(model.2, specs= pairwise~wool )
```

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


## $emmeans
##  wool emmean    SE df lower.CL upper.CL
##  A      3.32 0.072 48    3.17     3.46
##  B      3.17 0.072 48    3.02     3.31
##
## Results are averaged over the levels of: tension
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate    SE df t.ratio p.value
##  A - B       0.152 0.102 48 1.495   0.1415
##
## Results are averaged over the levels of: tension
## Results are given on the log (not the response) scale.
```

Using `emmeans`, we can see the wool effect difference between types B and A is -0.1522. We can calculate the mean number of log breaks for each wool type and take the difference by the following:

```r
warpbreaks %>%
  group_by(wool) %>%
  summarise( wool.means = mean(log(breaks)) ) %>%
  summarise( diff(wool.means) )
```

```
## `summarise()` ungrouping output (override with `.groups` argument)


## # A tibble: 1 x 1
##   `diff(wool.means)`
##               <dbl>
## 1            -0.152
```

In the unbalanced case taking the average of the cell means produces a different answer than taking the average of the data. The `emmeans` package chooses to take the average of the cell means.

6.6   Exercises

As we have developed much of the necessary theory, we are moving into exercises that emphasize modeling decisions and interpretation. As such, grading of exercises will move to emphasizing interpretation and justification of why modeling decisions were made. For any R output produced, be certain to discuss what is important. Feel free to not show exploratory work, but please comment that you considered. Furthermore, as students progress, necessary analysis steps will not be listed and students will be expected to appropriately perform the necessary work and comment as appropriate. These are leading to, at the end of the course, students being given a dataset and, with no prompts as to what is an appropriate analysis, told to model the data appropriately.

1. In the `faraway` package, the data set `rats` has data on a gruesome experiment that examined the time till death of 48 rats when they were subjected to three different types of poison administered in four different manners (which they called treatments). We are interested in assessing which poison works the fastest as well as which administration method is most effective.

   a. The response variable `time` needs to be transformed. To see this, we'll examine the diagnostic plots from the interaction model and there is clearly a problem with non-constant variance. We'll look at the Box-Cox family of transformations and see that y^{-1} is a reasonable transformation.

   ```
   data('rats', package='faraway')
   model <- lm( time ~ poison * treat, data=rats)
   # lindia::gg_diagnose(model)  # All the plots...
   lindia::gg_diagnose(model, plot.all=FALSE)[[4]]   # just resid vs fitted
   lindia::gg_boxcox(model)
   rats <- rats %>% mutate( speed = time^(-1) )
   ```

   b. Fit the interaction model using the transformed response. Create a graph of data and the predicted values. Visually assess if you think the interaction is significant.

   c. Perform an appropriate statistical test to see if the interaction is statistically significant.

   d. What do you conclude about the poisons and treatment (application) types?

2. In the `faraway` package, the dataset `butterfat` has information about the the percent of the milk was butterfat (more is better) taken from $n = 100$ cows. There are 5 different breeds of cows and 2 different ages. We are interested in assessing if `Age` and `Breed` affect the butterfat content

   a. Graph the data. Do you think an interaction model is justified?

b. Perform an appropriate set of tests to select a model for predicting `Butterfat`.

c. Discuss your findings.

3. In the `faraway` package, the dataset `alfalfa` has information from a study that examined the effect of seed inoculum, irrigation, and shade on alfalfa yield. This data has $n = 25$ observations.

a. Examine the help file for this dataset. Graph the data. What effects seem significant?

b. Consider the main effects model with all three predictor variables. Which effects are significant? *Using the model you ultimately select, examine the diagnostic plots. These all look fine, but it is useful to see examples where everything is ok.*

c. Consider the model with `shade` and `inoculum` and the interaction between the two. Examine the anova table. Why does R complain that the fit is perfect? *Hint: Think about the degrees of freedom of the model compared to the sample size.*

d. Discuss your findings and the limitations of your investigation based on data.

Chapter 7

Diagnostics

```r
library(ggfortify)   # for autoplot for lm objects
library(emmeans)     # emmeans for pairwise constrasts.
library(tidyverse)   # for dplyr, tidyr, ggplot2
```

We will be interested in analyzing whether or not our linear model is a good model and whether or not the data violate any of the assumptions that are required. In general we will be interested in three classes of assumption violations and our diagnostic measures might be able detect one or more of the following issues:

1. Unusual observations that contribute too much influence to the analysis. These few observations might drastically change the outcome of the model.

2. Model misspecification. Our assumption that $E[y] = X\beta$ might be wrong and we might need to include different covariates in the model to get a satisfactory result.

3. Error distribution. We have assumed that $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ but autocorrelation, heteroscedasticity, and non-normality might be present.

Often problems with one of these can be corrected by transforming either the explanatory or response variables.

7.1   Detecting Assumption Violations

Throughout this chapter I will use data created by Francis Anscombe that show how simple linear regression can be misused. In particular, these data sets will

show how our diagnostic measures will detect various departures from the model assumptions.

The data are available in R as a data frame **anscombe** and is loaded by default. The data consists of four datasets, each having the same linear regression $\hat{y} = 3 + 0.5\,x$ but the data are drastically different.

```r
# The anscombe dataset has 8 columns - x1,x2,x3,x4,y1,y2,y3,y4
# and I want it to have 3 columns - Set, X, Y
Anscombe <- rbind(
  data.frame(x=anscombe$x1, y=anscombe$y1, set='Set 1'),
  data.frame(x=anscombe$x2, y=anscombe$y2, set='Set 2'),
  data.frame(x=anscombe$x3, y=anscombe$y3, set='Set 3'),
  data.frame(x=anscombe$x4, y=anscombe$y4, set='Set 4'))

# order them by their x values, and add an index column
Anscombe <- Anscombe %>%
  group_by(set) %>%        # Every subsequent action happens by dataset
  arrange(x,y) %>%         # sort them on the x-values and if tied, by y-value
  mutate( index = 1:n() )  # give each observation within a set, an ID number

# Make a nice graph
ggplot(Anscombe, aes(x=x, y=y)) +
 geom_point() +
 facet_wrap(~set, scales='free') +
 stat_smooth(method="lm", formula=y~x, se=FALSE)
```
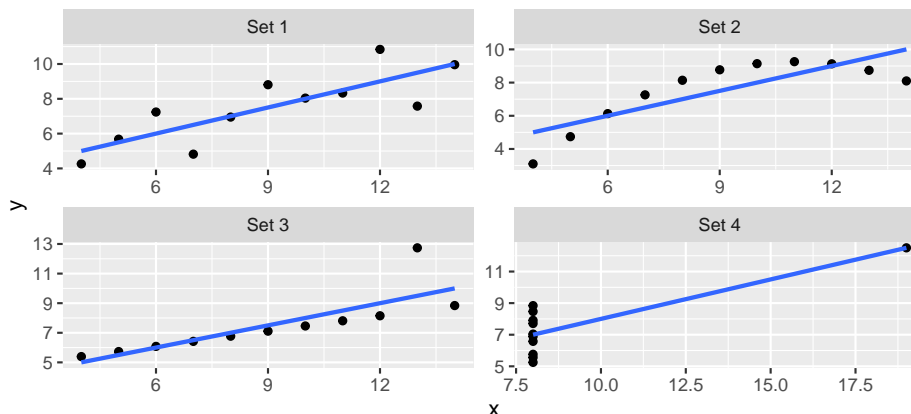

7.1.1   Measures of Influence

7.1.1.1   Standardized Residuals (aka Studentized)

Recall that we have

$$\hat{y} = X\hat{\beta}$$
$$= X\left(X^T X\right)^{-1} X^T y$$
$$= Hy$$

where the "Hat Matrix" is $H = X\left(X^T X\right)^{-1} X^T$ because we have $\hat{y} = Hy$. The elements of H can be quite useful in diagnostics. It can be shown that the variance of the $i$the residual is

$$Var\left(\hat{\epsilon}_i\right) = \sigma^2\left(1 - H_{ii}\right)$$

where H_{ii} is the ith element of the main diagonal of H. This suggests that I could rescale my residuals to

$$\hat{\epsilon}_i^* = \frac{\hat{\epsilon}_i}{\hat{\sigma}\sqrt{1 - H_{ii}}}$$

which, if the normality and homoscedasticity assumptions hold, should behave as a $N\left(0, 1\right)$ sample.

These rescaled residuals are called "studentized residuals", though R typically refers to them as "standardized". Since we have a good intuition about the scale of a standard normal distribution, the scale of standardized residuals will give a good indicator if normality is violated.

There are actually two types of studentized residuals, typically called *internal* and *external* among statisticians. The version presented above is the *internal* version which can be obtained using the R function `rstandard()` while the *external* version is available using `rstudent()`. Whenever you see R present standardized residuals, they are talking about internally studentized residuals. For sake of clarity, I will use the term *standardized* as well.

7.1.1.1.1 Example - Anscombe's set 3

For the third dataset, the outlier is the ninth observation with $x_9 = 13$ and $y_9 = 12.74$. We calculate the standardized residuals using the function `rstandard()` and plot them

```
Set3 <- Anscombe %>% filter(set == 'Set 3') # Just set 3
model <- lm(y ~ x, data=Set3)              # Fit the regression line
Set3$stdresid <- rstandard(model)          # rstandard() returns the standardized residuals

ggplot(Set3, aes(x=index, y=stdresid)) +   # make a plot
  geom_point() +
  labs(x='Observation Index',
       y='Standardized Residuals',
       title='Standardized Residuals vs Observation Index')
```

Standardized Residuals vs Observation Index
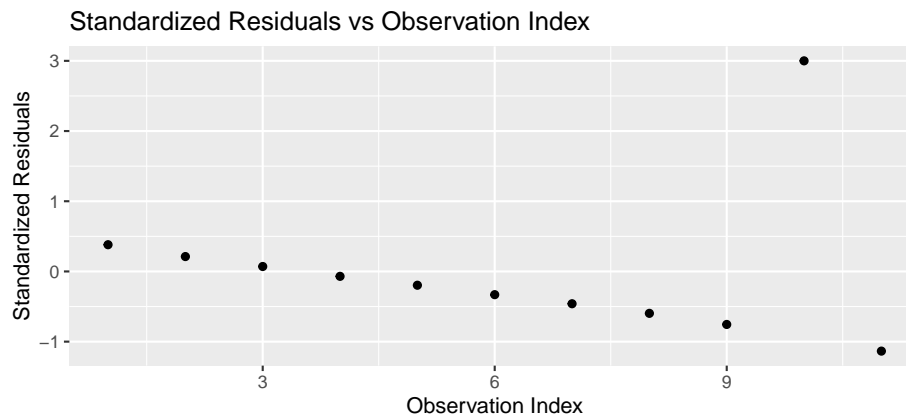
and we notice that the outlier residual is really big. If the model assumptions were true, then the standardized residuals should follow a standard normal distribution, and I would need to have hundreds of observations before I wouldn't be surprised to see a residual more than 3 standard deviations from 0.

7.1.1.2 Leverage

The extremely large standardized residual suggests that this data point is important, but we would like to quantify how important this observation actually is.

One way to quantify this is to look at the elements of H. Because

$$\hat{y}_i = \sum_{j=1}^{n} H_{ij} y_j$$

then the ith row of H is a vector of weights that tell us how influential a point y_j is for calculating the predicted value $\hat{y}_i$. If I look at just the main diagonal of H, these are how much weight a point has on its predicted value. As such, I can think of the H_{ii} as the amount of leverage a particular data point has on the regression line. It can be shown that the leverages must be $0 \le H_{ii} \le 1$ and that $\sum H_{ii} = p$.

```r
Set3 <- Anscombe %>% filter( set == 'Set 3')
Set4 <- Anscombe %>% filter( set == 'Set 4')

model3 <- lm(y ~ x, data = Set3 )
model4 <- lm(y ~ x, data = Set4 )

X <- model.matrix(model3)
H <- X %*% solve( t(X) %*% X) %*% t(X)

round(H, digits=2)
```

```
##        1     2     3    4    5    6    7    8     9    10    11
## 1   0.32  0.27  0.23 0.18 0.14 0.09 0.05 0.00 -0.05 -0.09 -0.14
## 2   0.27  0.24  0.20 0.16 0.13 0.09 0.05 0.02 -0.02 -0.05 -0.09
## 3   0.23  0.20  0.17 0.15 0.12 0.09 0.06 0.04  0.01 -0.02 -0.05
## 4   0.18  0.16  0.15 0.13 0.11 0.09 0.07 0.05  0.04  0.02  0.00
## 5   0.14  0.13  0.12 0.11 0.10 0.09 0.08 0.07  0.06  0.05  0.05
## 6   0.09  0.09  0.09 0.09 0.09 0.09 0.09 0.09  0.09  0.09  0.09
## 7   0.05  0.05  0.06 0.07 0.08 0.09 0.10 0.11  0.12  0.13  0.14
## 8   0.00  0.02  0.04 0.05 0.07 0.09 0.11 0.13  0.15  0.16  0.18
## 9  -0.05 -0.02  0.01 0.04 0.06 0.09 0.12 0.15  0.17  0.20  0.23
## 10 -0.09 -0.05 -0.02 0.02 0.05 0.09 0.13 0.16  0.20  0.24  0.27
## 11 -0.14 -0.09 -0.05 0.00 0.05 0.09 0.14 0.18  0.23  0.27  0.32
```

Fortunately there is already a function `hatvalues()` to compute these H_{ii} values for me. We will compare the leverages from Anscombe's set 3 versus set 4.

```
Set3 <- Set3 %>% mutate(leverage = hatvalues(model3))   # add leverage columns
Set4 <- Set4 %>% mutate(leverage = hatvalues(model4))

ggplot( rbind(Set3,Set4), aes(x=index, y=leverage) ) +
  geom_point() +
  facet_grid( . ~ set )
```
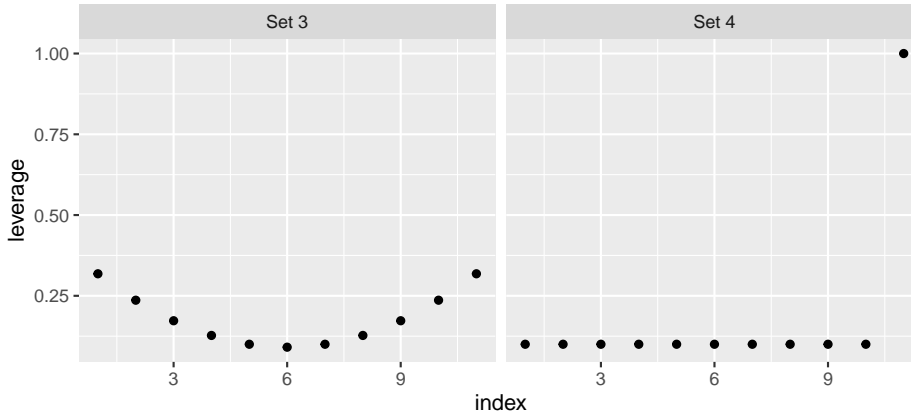

This leverage idea only picks out the *potential* for a specific value of x to be influential, but does not actually measure influence. It has picked out the issue with the fourth data set, but does not adequately address the outlier in set 3.

7.1.1.3   Cook's Distance

To attempt to measure the actual influence of an observation $\{y_i, x_i^T\}$ on the linear model, we consider the effect on the regression if we removed the observation

and fit the same model. Let
$$\hat{y} = X\hat{\beta}$$
be the vector of predicted values, where $\hat{\beta}$ is created using all of the data, and $\hat{y}_{(i)} = X\hat{\beta}_{(i)}$ be the vector of predicted values where $\hat{\beta}_{(i)}$ was estimated using all of the data except the ith observation. Letting p be the number of β_j parameters as usual we define Cook's distance of the ith observation as

$$D_i = \frac{\left(\hat{y} - \hat{y}_{(i)}\right)^T \left(\hat{y} - \hat{y}_{(i)}\right)}{p\hat{\sigma}^2} = \frac{\sum_{j=1}^{n}(\hat{y}_j - (\hat{y}_{(i)})_j)^2}{p\hat{\sigma}^2}$$

which boils down to saying if the predicted values have large changes when the ith element is removed, then the distance is big. It can be shown that this formula can be simplified to

$$D_i = \frac{\hat{\epsilon}_i^* H_{ii}}{p\left(1 - H_{ii}\right)}$$

which expresses Cook's distance in terms of the ith studentized residual and the ith leverage.

Nicely, the R function `cooks.distance()` will calculate Cook's distance.

```
Set3 <- Set3 %>% mutate(cooksd = cooks.distance(model3))
Set4 <- Set4 %>% mutate(cooksd = cooks.distance(model4))

# Note: The high leverage point in set 4 has a Cook's distance of Infinity.
ggplot(rbind(Set3,Set4), aes(x=index, y=cooksd)) +
  geom_point() +
  facet_grid(. ~ set) +
  labs(y="Cook's Distance")
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```
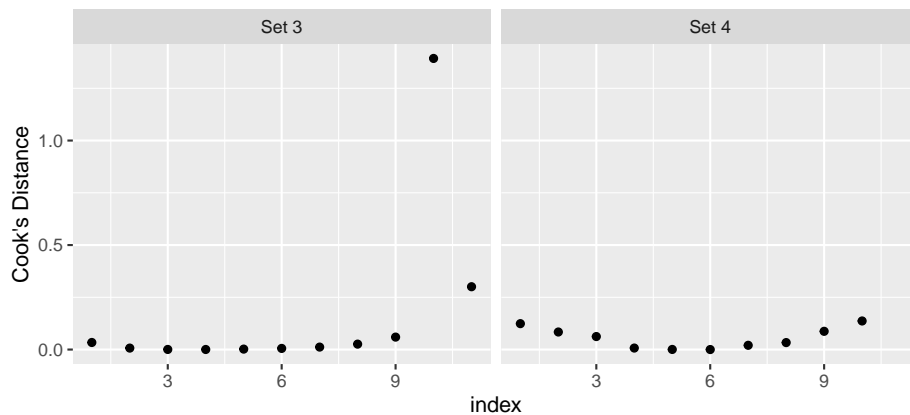
Some texts will give a rule of thumb that points with Cook's distances greater than 1 should be considered influential, while other books claim a reasonable rule of thumb is $4/(n - p - 1)$ where n is the sample size, and p is the number of parameters in β. My take on this, is that you should look for values that are highly different from the rest of your data.

7.1.2 Diagnostic Plots

After fitting a linear model in R, you have the option of looking at diagnostic plots that help to decide if any assumptions are being violated. We will step through each of the plots that are generated by the function `plot(model)` or using `ggplot2` using the package `ggfortify`.

In the package `ggfortify` there is a function that will calculate the diagnostics measures and add them to your dataset. This will simplify our graphing process.

```
Set1 <- Anscombe %>% filter(set == 'Set 1')
model <- lm( y ~ x, data=Set1)
Set1 <- fortify(model)  # add diagnostic measures to the dataset
Set1 %>% round(digits=3) # show the dataset nicely
```

```
##          y  x  .hat .sigma .cooksd .fitted .resid .stdresid
## 1     4.26  4 0.318  1.273   0.123   5.000 -0.740    -0.725
## 2     5.68  5 0.236  1.310   0.004   5.501  0.179     0.166
## 3     7.24  6 0.173  1.220   0.127   6.001  1.239     1.102
## 4     4.82  7 0.127  1.147   0.154   6.501 -1.681    -1.455
## 5     6.95  8 0.100  1.311   0.000   7.001 -0.051    -0.043
## 6     8.81  9 0.091  1.218   0.062   7.501  1.309     1.110
## 7     8.04 10 0.100  1.312   0.000   8.001  0.039     0.033
## 8     8.33 11 0.127  1.310   0.002   8.501 -0.171    -0.148
## 9    10.84 12 0.173  1.100   0.279   9.001  1.839     1.635
## 10    7.58 13 0.236  1.056   0.489   9.501 -1.921    -1.778
## 11    9.96 14 0.318  1.311   0.000  10.001 -0.041    -0.041
```

7.1.2.1 Residuals vs Fitted

In the simple linear regression the most useful plot to look at was the residuals versus the x-covariate, but we also saw that this was similar to looking at the residuals versus the fitted values. In the general linear model, we will look at the residuals versus the fitted values or possibly the studentized residuals versus the fitted values.

7.1.2.1.1   Polynomial relationships
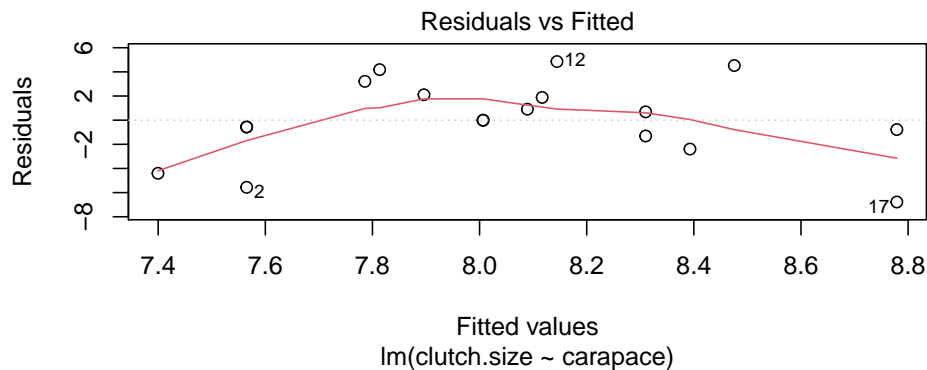
To explore how this plot can detect non-linear relationships between y and x, we will examine a data set from Ashton et al. (2007) that relates the length of a tortoise's carapace to the number of eggs laid in a clutch. The data are

```
Eggs <- data.frame(
  carapace    = c(284,290,290,290,298,299,302,306,306,
                  309,310,311,317,317,320,323,334,334),
  clutch.size = c(3,2,7,7,11,12,10,8,8,
                  9,10,13,7,9,6,13,2,8))
ggplot(Eggs, aes(x=carapace, y=clutch.size)) +
  geom_point()
```
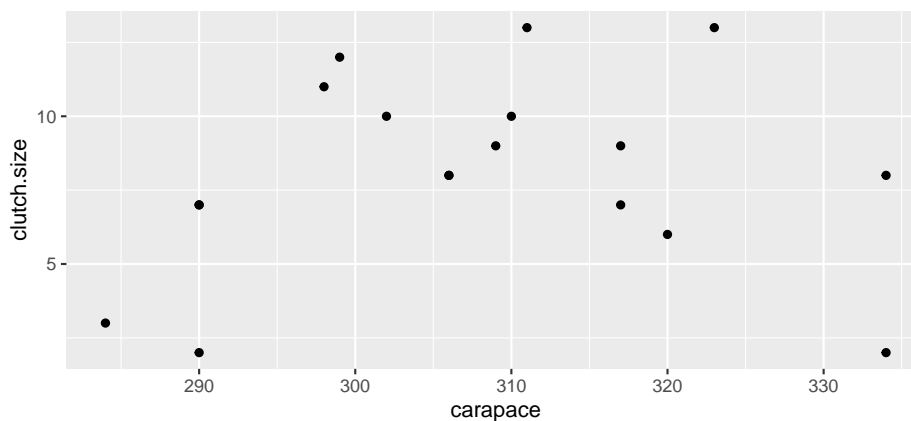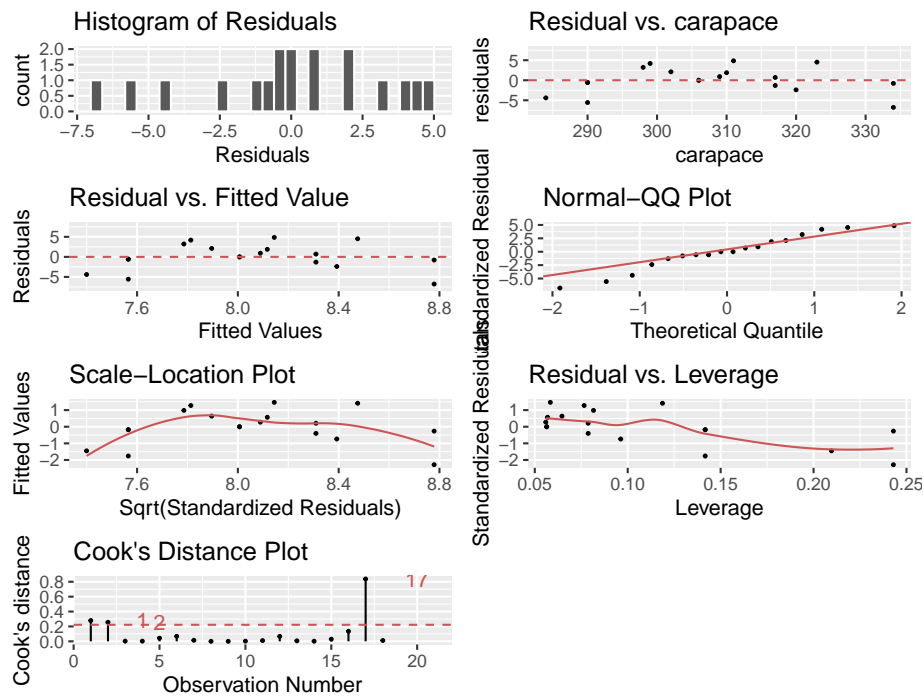

Looking at the data, it seems that we are violating the assumption that a linear model is appropriate, but we will fit the model anyway and look at the residual graph.

```
model <- lm( clutch.size ~ carapace, data=Eggs )
plot(model, which=1)        # Base R Function: which=1 tells R to only make the first pl
```
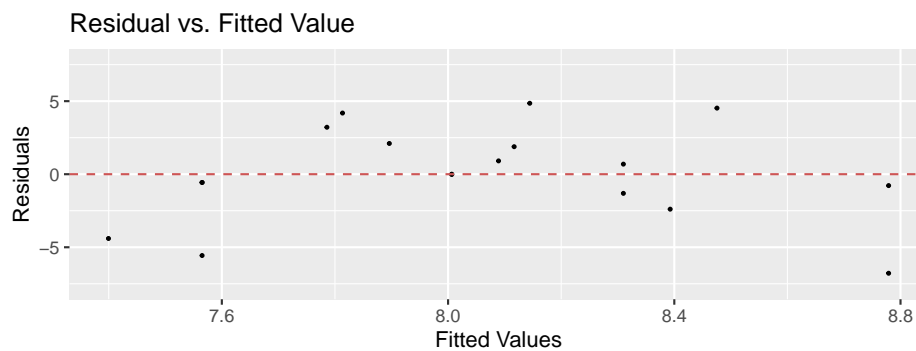
```
lindia::gg_diagnose(model)                          # using lindia package
```

Histogram of Residuals

Residual vs. carapace

Residual vs. Fitted Value

Normal–QQ Plot

Scale–Location Plot

Residual vs. Leverage

Cook's Distance Plot

```
lindia::gg_diagnose(model, plot.all=FALSE)[[3]]   # using lindia package
```

Residual vs. Fitted Value

```
autoplot(model, which=1)   # same plot using ggplot2 and ggfortify package
```

Residuals vs Fitted
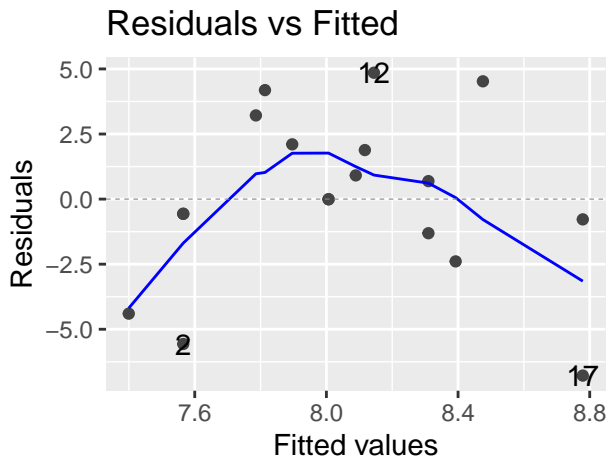
The blue curves going through the plot is a smoother of the residuals. Ideally this should be a flat line and I should see no trend in this plot. Clearly there is a quadratic trend as larger tortoises have larger clutch sizes until some point where the extremely large tortoises start laying fewer (perhaps the extremely large tortoises are extremely old as well). To correct for this, we should fit a model that is quadratic in `carapace length`. We will create a new covariate, `carapace.2`, which is the square of the carapace length and add it to the model.

In general I could write the quadratic model as

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i$$

and note that my model is still a linear model with respect to covariates x and x^2 because I can still write the model as

$$y = X\beta + \epsilon$$

$$= \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

```
# add a new column that is carapace^2
Eggs2 <- Eggs %>% mutate( carapace.2 = carapace^2 )
model <- lm( clutch.size ~ carapace + carapace.2,    data=Eggs2 )

# make R do it inside the formula... convenient
model <- lm( clutch.size ~ carapace + I(carapace^2), data=Eggs )
```

```r
# Fit an arbitrary degree polynomial, I recommend this method for fitting the model!
model <- lm( clutch.size ~ poly(carapace, 2),          data=Eggs )

# If you use poly() in the formula, you must use 'data=' here,
# otherwise you can skip it and R will do the right thing.
autoplot(model, which=1, data=Eggs)
```
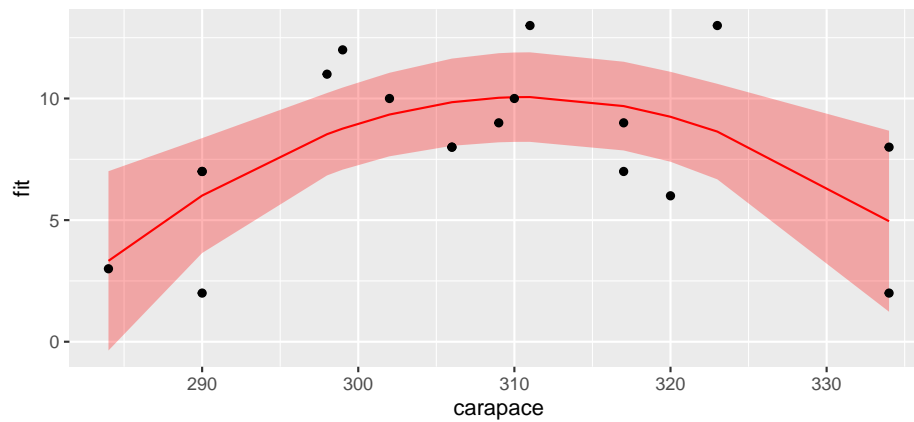
Residuals vs Fitted
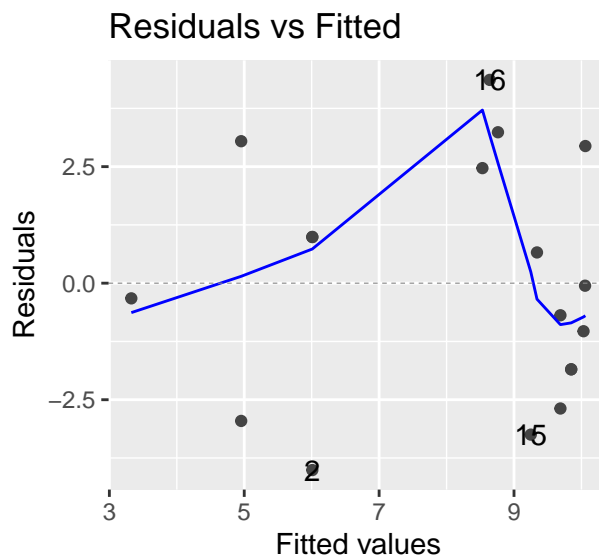
Now our residual plot versus fitted values does not show any trend, suggesting that the quadratic model is fitting the data well. Graphing the original data along with the predicted values confirms this.

```r
# add the fitted and CI lwr/upr columns to my dataset
Eggs <- Eggs %>%
  select( -matches('fit'), -matches('lwr'), -matches('upr') ) %>%
  cbind( predict(model, interval='confidence') )

ggplot(Eggs, aes(x=carapace)) +
  geom_ribbon( aes(ymin=lwr, ymax=upr), fill='red', alpha=.3) +
  geom_line(aes(y=fit), color='red') +
  geom_point(aes(y=clutch.size))
```

7.1.2.1.2 Heteroskedasticity

The plot of residuals versus fitted values can detect heteroskedasticity (non-constant variance) in the error terms.

To illustrate this, we turn to another dataset in the Faraway book. The dataset `airquality` uses data taken from an environmental study that measured four variables, ozone, solar radiation, temperature and wind speed for 153 consecutive days in New York. The goal is to predict the level of ozone using the weather variables.

We first graph all pairs of variables in the dataset.

```r
data(airquality)
# pairs(~ Ozone + Solar.R + Wind + Temp, data=airquality)
airquality %>% select( Solar.R, Wind, Temp, Ozone) %>%
  GGally::ggpairs()
```
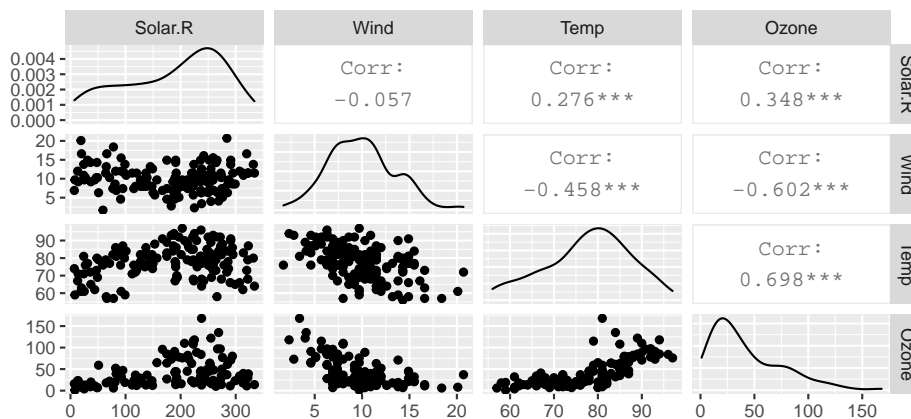
and notice that ozone levels are positively correlated with solar radiation and temperature, and negatively correlated with wind speed. A linear relationship with wind might be suspect as is the increasing variability in the response to high temperature. However, we don't know if those trends will remain after fitting the model, because there is some covariance among the predictors.

```
model <- lm(Ozone ~ Solar.R + Wind + Temp, data=airquality)
autoplot(model, which=1)
```

Residuals vs Fitted
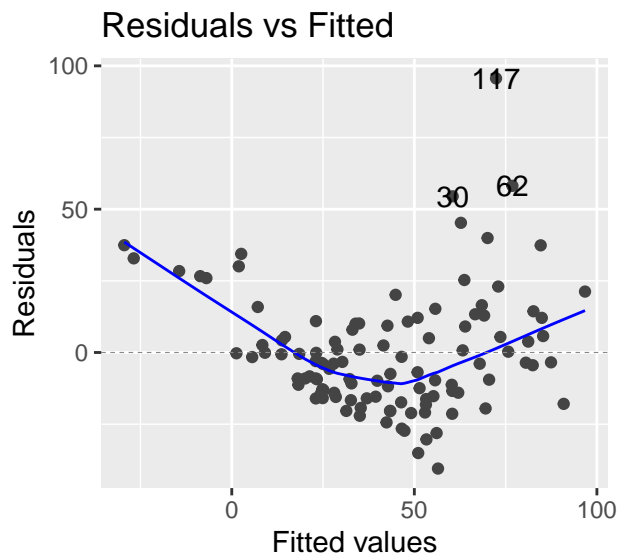
As we feared, we have both a non-constant variance and a non-linear relationship. A transformation of the y variable might be able to fix our problem.

7.1.2.2 QQplots

If we are taking a sample of size $n = 10$ from a standard normal distribution, then I should expect that the smallest observation will be negative. Intuitively, you would expect the smallest observation to be near the 10th percentile of the standard normal, and likewise the second smallest should be near the 20th percentile.

This idea needs a little modification because the largest observation cannot be near the 100th percentile (because that is ∞). So we'll adjust the estimates to still be spaced at $(1/n)$ quantile increments, but starting at the $0.5/n$ quantile instead of the $1/n$ quantile. So the smallest observation should be near the 0.05 quantile, the second smallest should be near the 0.15 quantile, and the largest observation should be near the 0.95 quantile. I will refer to these as the *theoretical quantiles.*

Std Normal distribution
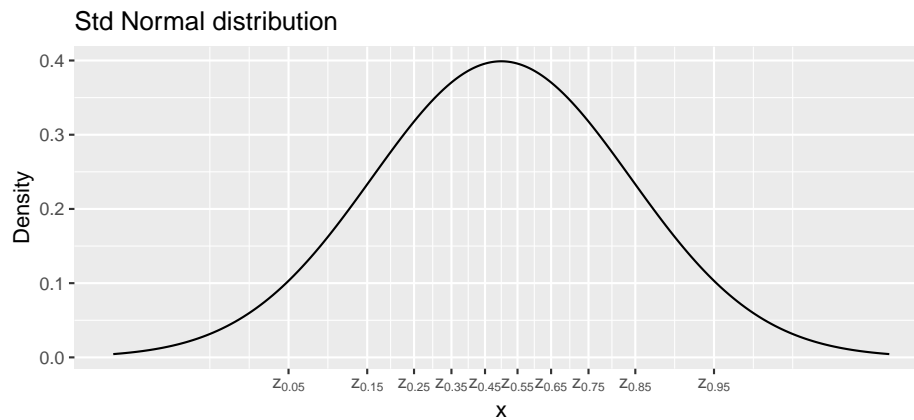
I can then graph the theoretical quantiles vs my observed values and if they lie on the 1-to-1 line, then my data comes from a standard normal distribution.

```r
set.seed(93516)  # make random sample in the next code chunk consistant run-to-run

n <- 10
data <- data.frame( observed = rnorm(n, mean=0, sd=1) ) %>%
  arrange(observed) %>%
  mutate( theoretical = qnorm( (1:n -.5)/n ) )

ggplot(data, aes(x=theoretical, y=observed) ) +
  geom_point() +
  geom_abline( intercept=0, slope=1,  linetype=2, alpha=.7) +
  labs(main='Q-Q Plot: Observed vs Normal Distribution')
```
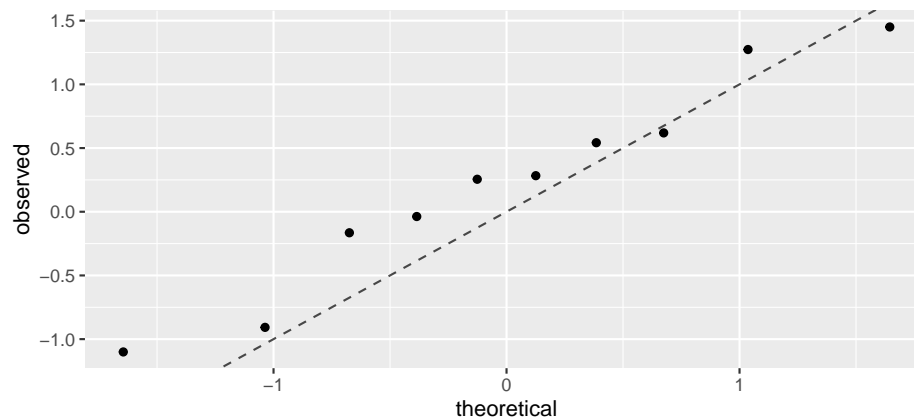

In the context of a regression model, we wish to look at the residuals and see if there are obvious departures from normality. Returning to the air quality example, R will calculate the qqplot for us.

```
model <- lm(Ozone ~ Solar.R + Wind + Temp, data=airquality)
autoplot(model, which=2)
```
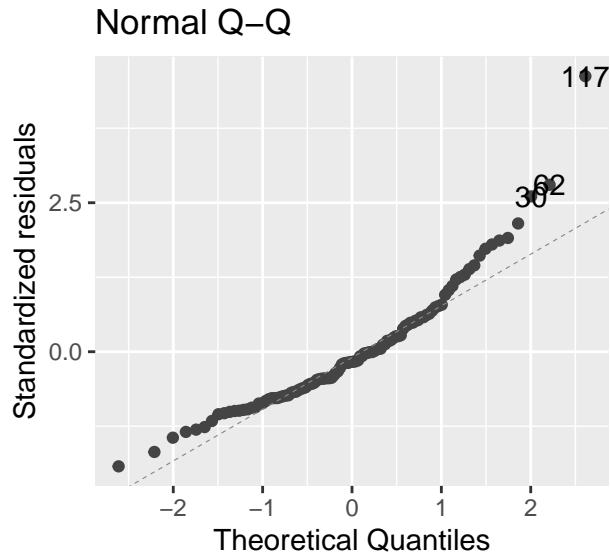

In this case, we have a large number of residuals that are bigger than I would expect them to be based on them being from a normal distribution. We could further test this using the Shapiro-Wilks test and compare the standardized residuals against a $N(0, 1)$ distribution.

```
shapiro.test( rstandard(model) )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(model)
## W = 0.9151, p-value = 2.819e-06
```

The tail of the distribution of observed residuals is *far* from what we expect to see.

7.1.2.3 Scale-Location Plot

This plot is a variation on the fitted vs residuals plot, but the y-axis uses the square root of the absolute value of the standardized residuals. Supposedly this makes detecting increasing variance easier to detect, but I'm not convinced.

7.1.2.4   Residuals vs Leverage (plus Cook's Distance)

This plot lets the user examine the which observations have a high potential for being influential (i.e. high leverage) versus how large the residual is. Because Cook's distance is a function of those two traits, we can also divide the graph up into regions by the value of Cook's Distance.

Returning to Anscombe's third set of data, we see

```
model3 <- lm(y ~ x, data=Set3)
autoplot(model3, which=5)
```
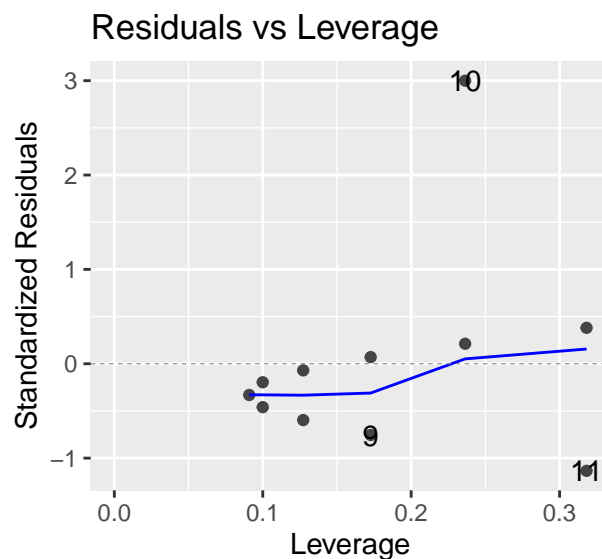

that one data point (observation 10) has an extremely large standardized residual. This is one plot where I prefer what the base graphics in R does compared to the `ggfortify` version. The base version of R adds some contour lines that mark where the contours of where Cook's distance is 1/2 and 1.

```
plot(model3, which=5)
```

7.2 Exercises
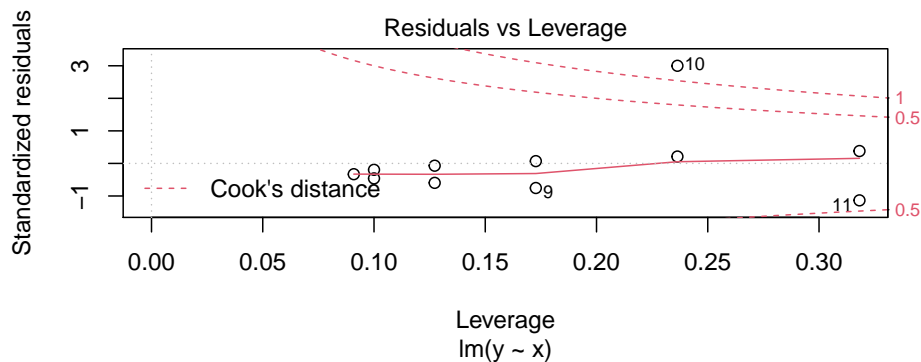
1. In the ANCOVA chapter, we examined the relationship on dose of vitamin C on guinea pig tooth growth based on how the vitamin was delivered (orange juice vs a pill supplement).

    a. Load the `ToothGrowth` data which is pre-loaded in base R.
    b. Plot the data with `dose` level on the x-axis and tooth length growth (`len`) on the y-axis. Color the points by supplement type (`supp`).
    c. Is the data evenly distributed along the x-axis? Comment on the wisdom of using this design.
    d. Fit a linear model to these data and examine the diagnostic plots. What stands out to you?
    e. Log-transform the dose variable and repeat parts (c) and (d). Comment on the effect of the log transformation.

2. The dataset `infmort` in the `faraway` package has information about infant mortality from countries around the world. Be aware that this is a old data set and does not necessarily reflect current conditions. More information about the dataset can be found using `?faraway::infmort`. We will be interested in understanding how infant mortality is predicted by per capita income, world region, and oil export status.

    a. Plot the relationship between income and mortality. This can be done using the command

    ```
    data('infmort', package='faraway')
    pairs(mortality ~., data=infmort)
    ```

    What do you notice about the relationship between mortality and income?

    b. Fit a linear model without any interaction terms with all three covariates as predictors of infant mortality. Examine the diagnostic plots. What stands out?

    c. Examine the pairs plot with log(mortality), income, and log(income). Which should be used in our model, `income` or `log(income)`?

3. The dataset`pressure` in the `datasets` package gives emperical measurements of the vapor pressure of mercury at various temperatures. Fit a model with pressure as the response and temperature as the predictor using transformations to obtain a good fit. Feel free to experiment with what might be considered a ridiculously complicated model with a high degree polynomial. These models can most easily be fit using the `poly(x, degree=p)` function in the formula specification where you swap out the covariate `x` and polynomial degree `p`.

    a. Document your process of building your final model. Do not show graphs or computer output that is not relevant to your decision or that you do not wish to comment on.

    b. Comment on the interpretability of your (possibly ridiculously complicated) model. Consider a situation where I'm designing a system where it is easy to measure temperature, but I am unable to measure pressure. Could we get by with just measuring temperature? What if I didn't know the Clausius-Clapeyron equation (which describes vapor pressure at various temperatures) and I'm trying to better understand how temperature and pressure are related?

4. We will consider the relationship between income and race using a subset of employed individuals from the American Community Survey.

    a. Load the `EmployedACS` dataset from the `Lock5Data` package.

    b. Create a box plot showing the relationship between `Race` and `Income`.

    c. Fit an ANOVA model to this data and consider the diagnostic plots for the residuals. What do you notice?

Chapter 8

Data Transformations

```r
library(ggfortify)   # for autoplot for lm objects
library(emmeans)     # emmeans for pairwise constrasts.
library(tidyverse)   # for dplyr, tidyr, ggplot2
```

Transformations of the response variable and/or the predictor variables can drastically improve the model fit and can correct violations of the model assumptions. We might also create new predictor variables that are functions of existing variables. These include quadratic and higher order polynomial terms and interaction terms.

Often we are presented with data and we would like to fit a linear model to the data. Unfortunately the data might not satisfy all of the assumptions of a linear model. For the simple linear model

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

where $\epsilon_i \overset{iid}{\sim} N\left(0, \sigma^2\right)$, the necessary assumptions are (in order of importance):

1. The model contains all the appropriate covariates and no more.
2. Independent errors. *(Hard to check this one!)*
3. Errors have constant variance, no matter what the x-value (or equivalently the fitted value)
4. Errors are normally distributed

In general, a transformation of the response variable can be used to address the 2nd and 3rd assumptions, and adding new covariates to the model will be how to address deficiencies of assumption 4. Because of the interpretability properties we will develop here, log() transformations are very popular, if they are useful.

8.1   A review of $\log(x)$ and e^x

One of the most common transformations that is used on either the response y or the covariates x is the log() function. In this next section we will consider log() with base e. However, if you prefer $\log_2()$ or $\log_{10}$ you may substitute e with 2 or 10 everywhere.

In primary school you might have learned that the log() function looks like this:
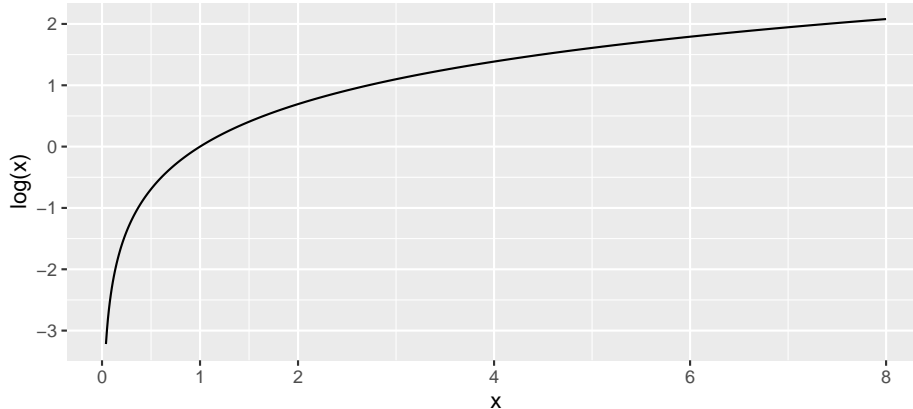
Critical aspects to notice about $\log(x)$:

1. As $x \to 0$, $\log(x) \to -\infty$.
2. At $x = 1$ we have $log(x = 1) = 0$.
3. As $x \to \infty$, $\log(x) \to \infty$ as well, but at a *much* slower rate.
4. Even though $log(x)$ is only defined for $x > 0$, the result can take on any real value, positive or negative.

The inverse function of $\log(x)$ is $e^x = \exp(x)$, where $e = 2.71828\ldots$ which looks
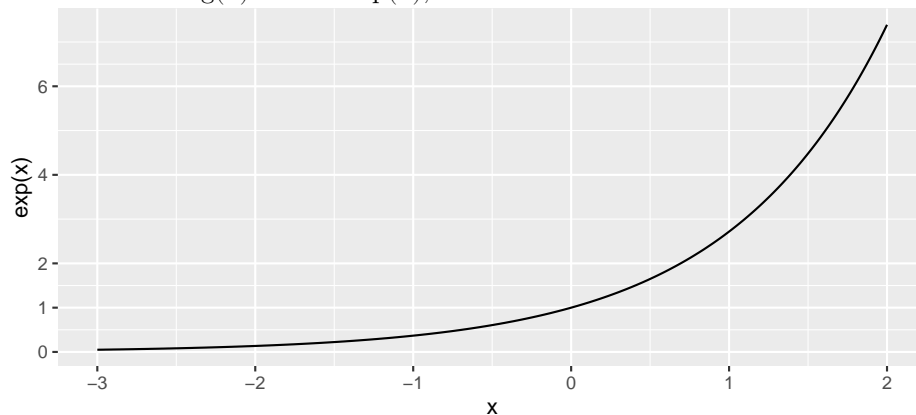
like this:

Critical aspects to notice about e^x:

1. as $x \to -\infty$, $e^x \to 0$.
2. At $x = 0$ we have $e^0 = 1$.
3. as $x \to \infty$, $e^x \to \infty$ as well, but at a *much* faster rate.
4. The function e^x can be evaluated for any real number, but the result is always > 0.

Finally we have that e^x and $log(x)$ are inverse functions of each other by the following identity:

$$x = \log(e^x)$$

and

$$x = e^{\log(x)} \quad \text{if } x > 0$$

Also it is important to note that the log function has some interesting properties in that it makes operations "1-operation easier".

$$\log(a^b) = b \log a$$
$$\log\left(\frac{a}{b}\right) = \log a - \log b$$
$$\log(ab) = \log a + \log b$$

One final aspect of exponents that we will utilize is that

$$e^{a+b} = e^a e^b$$

The reason we like using a log() transformation is that it acts differently on large values than small. In particular for $x > 1$ we have that $log(x)$ makes all of the smaller, but the transformation on big values of x is more extreme. Consider the following, where most of the x-values are small, but we have a few that are quite large. Those large values will have extremely high leverage and we'd like to reduce that.
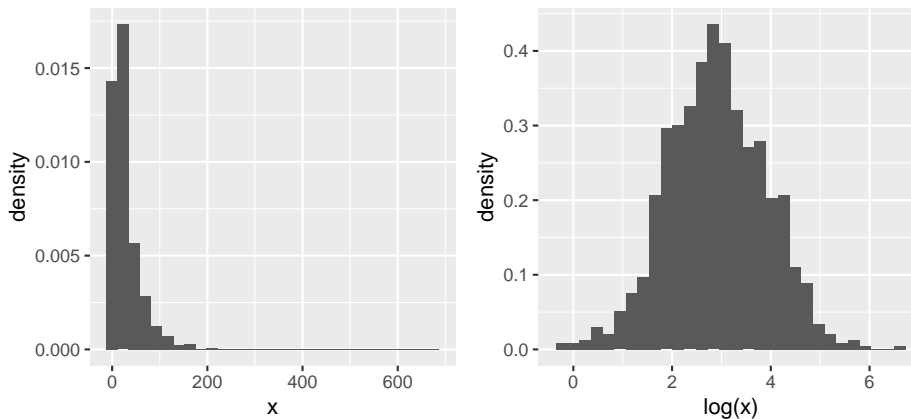
8.2   Transforming the Response

When the normality or constant variance assumption is violated, sometimes it is possible to transform the response to satisfy the assumption. Often times count data is analyzed as `log(count)` and weights are analyzed after taking a square root or cube root transform. Statistics involving income or other monetary values are usually analyzed on the log scale so as to reduce the leverage of high income observations.
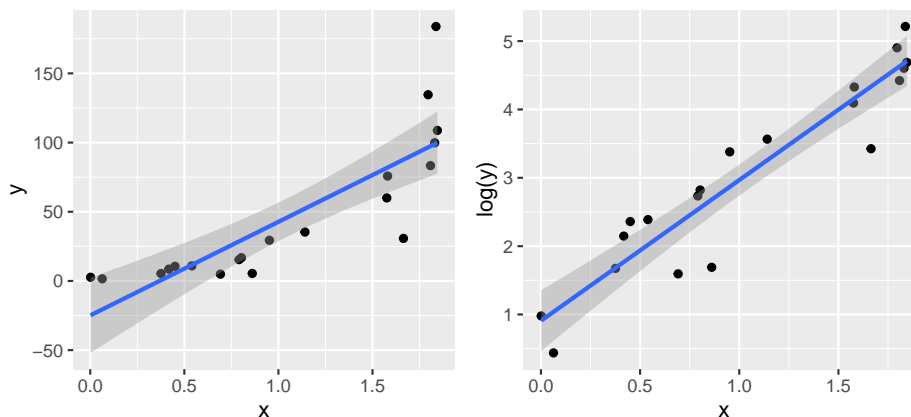
Clearly the model fit to the log transformed y-variable is a much better regression model. However, I would like to take the regression line and confidence interval back to the original y-scale. This is allowed by doing the inverse function $e^{\log(\hat{y})}$.

For example if we fit a linear model for income (y) based on the amount of schooling the individual has received (x). In this case, I don't really want to make predictions on the $\log(y)$ scale, because (almost) nobody will understand magnitude difference between predicting 5 vs 6.

Suppose the model is

$$\log y = \beta_0 + \beta_1 x + \epsilon$$

then we might want to give a prediction interval for an x_0 value. The predicted $log(income)$ value is

$$\log(\hat{y}_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

and we could calculate the appropriate predicted income as

$$\hat{y}_0 = e^{\hat{\beta}_0 + \hat{\beta}_1 x_0} = e^{log(\hat{y}_0)}$$

Likewise if we had a confidence interval or prediction interval for $\log(\hat{y}_0)$ of the form (l, u) then the appropriate interval for $\hat{y}_0$ is (e^l, e^u). Notice that while (l, u) might be symmetric about $\log(\hat{y}_0)$, the back-transformed interval is not symmetric about $\hat{y}_0$.

```
model <- lm( log(y) ~ x, data)
data <- data %>%
  select( -matches('(fit|lwr|upr)'))  %>%
  cbind( predict(model, newdata=., interval='confidence') )
data <- data %>% mutate(
  fit = exp(fit),
  lwr = exp(lwr),
  upr = exp(upr))

ggplot(data, aes(x=x)) +
  geom_ribbon( aes(ymin=lwr, ymax=upr), alpha=.6 ) +
  geom_line( aes(y=fit), color='blue' ) +
  geom_point( aes(y=y) ) +
  labs(y='y')
```
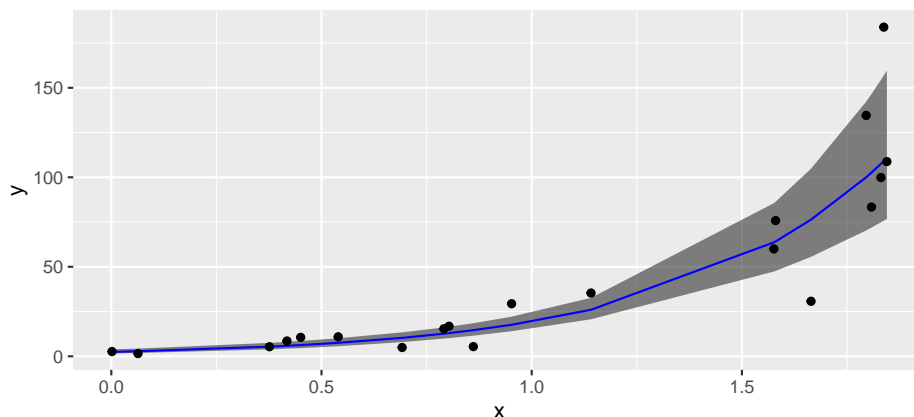

This back transformation on the $\hat{y}$ values will be acceptable for any 1-to-1 transformation we use, not just $\log(y)$.

Unfortunately the interpretation of the regression coefficients $\hat{\beta}_0$ and $\hat{\beta}_1$ on the un-transformed scale becomes more complicated. This is a very serious difficulty and might sway a researcher from transforming their data.

8.2.1 Box-Cox Family of Transformations

The Box-Cox method is a popular way of determining what transformation to make. It is intended for responses that are strictly positive (because $\log 0 = -\infty$ and the square root of a number gives complex numbers, which we don't know how to address in regression). The transformation is defined as

$$g(y) = \begin{cases} \frac{y^\lambda - 1}{\lambda} & \lambda \neq 0 \\ \log y & \lambda = 0 \end{cases}$$

This transformation is a smooth family of transformations because

$$\lim_{\lambda \to 0} \frac{y^{\lambda} - 1}{\lambda} = \log y$$

In the case that $\lambda \neq 0$, then a researcher will usually use the simpler transformation y^{λ} because the subtraction and division does not change anything in a non-linear fashion. Thus for purposes of addressing the assumption violations, all we care about is the y^{λ} and prefer the simpler (i.e. more interpretable) transformation.

Finding the best transformation can be done by adding the λ parameter to the model and finding the value that maximizes the log-likelihood function. Fortunately, we don't have to do this by hand, as the function `boxcox()` in the `MASS` library will do all the heavy calculation for us.

```
data(gala, package='faraway')
g <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent, data=gala)

# I don't like loading the MASS package because it includes a select() function
# that fights with dplyr::select(), so whenever I use a function in the MASS
# package, I just call it using the package::function() naming.
#
# #MASS::boxcox(g, lambda=seq(-2,2, by=.1))  # Set lambda range manually...
MASS::boxcox( g )    # With default lambda range.
```
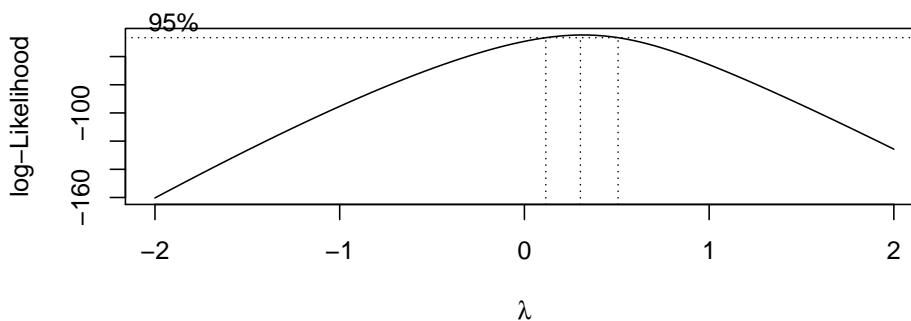

The optimal transformation for these data would be $y^{1/4} = \sqrt[4]{y}$ but that is an extremely uncommon transformation. Instead we should pick the nearest "standard" transformation which would suggest that we should use either the $\log y$ or $\sqrt{y}$ transformation.

Thoughts on the Box-Cox transformation:

1. In general, I prefer to using a larger-than-optimal model when picking a transformation and then go about the model building process. After a suitable model has been chosen, I'll double check the my transformation was appropriate given the model that I ended up with.

2. Outliers can have a profound effect on this method. If the "optimal" transformation is extreme ($\lambda = 5$ or something silly) then you might have to remove the outliers and refit the transformation.
3. If the range of the response y is small, then the method is not as sensitive.
4. These are not the only possible transformations. For example, for binary data, the `logit` and `probit` transformations are common. In classical non-parametric statistics, we take a rank transformation to the y-values.

8.3 Transforming the predictors

8.3.1 Polynomials of a predictor

Perhaps the most common transformation to make is to make a quadratic function in x. Often the relationship between x and y follows a curve and we want to fit a quadratic model

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x + \hat{\beta}_2 x^2$$

and we should note that this is still a linear model because $\hat{y}$ is a linear function of x and x^2. As we have already seen, it is easy to fit the model. Adding the column of x^2 values to the design matrix does the trick.

The difficult part comes in the interpretation of the parameter values. No longer is $\hat{\beta}_1$ the increase in y for every one unit increase in x. Instead the three parameters in my model interact in a complicated fashion. For example, the peak of the parabola is at $-\hat{\beta}_1/2\hat{\beta}_2$ and whether the parabola is cup shaped vs dome shaped and its steepness is controlled by $\hat{\beta}_2$. Because my geometric understanding of degree q polynomials relies on have all factors of degree q or lower, whenever I include a covariate raised to a power, I should include all the lower powers as well.

8.3.2 Log and Square Root of a predictor

Often the effect of a covariate is not linearly related to response, but rather some function of the covariate. For example the area of a circle is not linearly related to its radius, but it is linearly related to the radius squared.

$$Area = \pi r^2$$

Similar situations might arise in biological settings, such as the volume of conducting tissue being related to the square of the diameter. Or perhaps an animals metabolic requirements are related to some power of body length. In sociology, it is often seen that the utility of, say, \$1000 drops off in a logarithmic fashion according to the person's income. To a graduate student, \$1K is a big deal, but to a corporate CEO, \$1K is just another weekend at the track.

Making a log transformation on any monetary covariate, might account for the non-linear nature of "utility".

Picking a good transformation for a covariate is quite difficult, but most fields of study have spent plenty of time thinking about these issues. When in doubt, look at scatter plots of the covariate vs the response and ask what transformation would make the data fall onto a line?

8.3.3   Galapagos Example

To illustrate how to add a transformation of a predictor to a linear model in R, we will consider the Galapagos data in `faraway`.

```
data('gala', package='faraway')
# look at all the scatterplots
gala %>%
  mutate(LogSpecies = log(Species)) %>%
  dplyr::select(LogSpecies, Area, Elevation, Nearest, Scruz, Adjacent) %>%
  GGally::ggpairs(upper=list(continuous='points'), lower=list(continuous='cor'))
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```
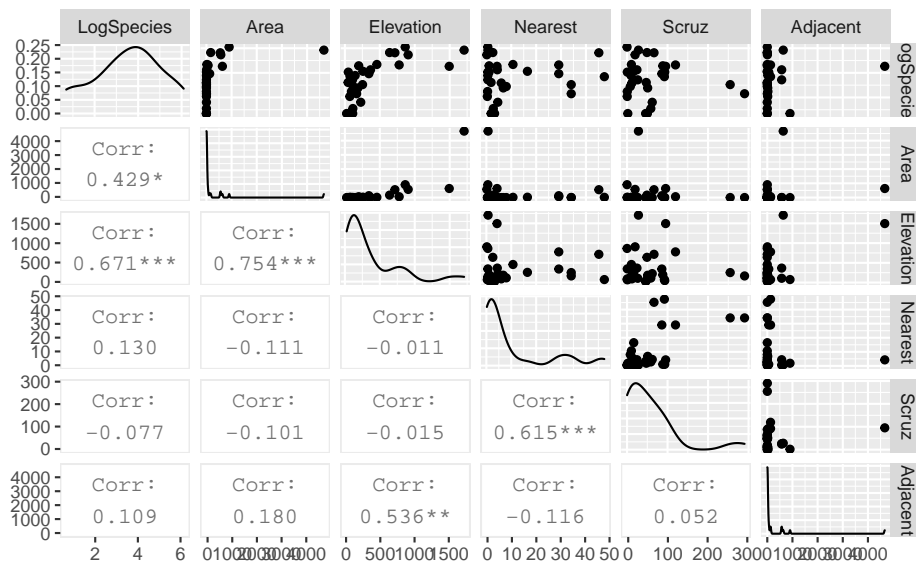

Looking at these graphs, I think I should definitely transform `Area` and `Adjacent`, and I wouldn't object to doing the same to `Elevation`, `Nearest` and

`Scruz`. Given the high leverages, a log transformation should be a good idea. One problem is that $\log(0) = -\infty$. A quick look at the data set summary:

```
gala %>%
  dplyr::select(Species, Area, Elevation, Nearest,Scruz, Adjacent) %>%
  summary()
```

```
##     Species              Area              Elevation            Nearest
##  Min.   :  2.00   Min.   :   0.010   Min.   :  25.00   Min.   : 0.20
##  1st Qu.: 13.00   1st Qu.:   0.258   1st Qu.:  97.75   1st Qu.: 0.80
##  Median : 42.00   Median :   2.590   Median : 192.00   Median : 3.05
##  Mean   : 85.23   Mean   : 261.709   Mean   : 368.03   Mean   :10.06
##  3rd Qu.: 96.00   3rd Qu.:  59.237   3rd Qu.: 435.25   3rd Qu.:10.03
##  Max.   :444.00   Max.   :4669.320   Max.   :1707.00   Max.   :47.40
##      Scruz             Adjacent
##  Min.   :  0.00   Min.   :   0.03
##  1st Qu.: 11.03   1st Qu.:   0.52
##  Median : 46.65   Median :   2.59
##  Mean   : 56.98   Mean   : 261.10
##  3rd Qu.: 81.08   3rd Qu.:  59.24
##  Max.   :290.20   Max.   :4669.32
```

reveals that `Scruz` has a zero value, and so a log transformation will result in a $-\infty$. So, lets take the square root of `Scruz`

```
gala %>%
  mutate(LogSpecies = log(Species), LogElevation=log(Elevation), LogArea=log(Area), LogNearest=lo
          SqrtScruz=sqrt(Scruz), LogAdjacent=log(Adjacent)) %>%
  dplyr::select(LogSpecies, LogElevation, LogArea, LogNearest, SqrtScruz, LogAdjacent) %>%
  GGally::ggpairs(upper=list(continuous='points'), lower=list(continuous='cor'))
```
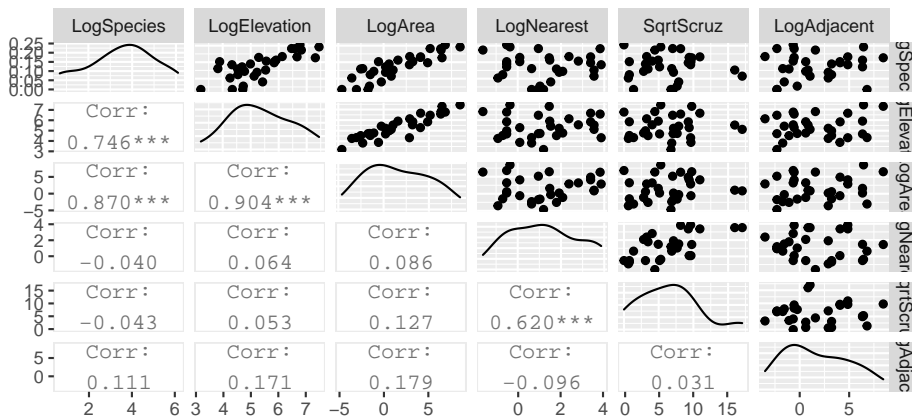
Looking at these graphs, it is clear that `log(Elevation)` and `log(Area)` are
highly correlated and we should probably have one or the other, but not both
in the model.

```
m.c <- lm(log(Species) ~ log(Area) + log(Nearest) + sqrt(Scruz) + log(Adjacent), data=
summary(m.c)$coefficients %>% round(digits=3) # more readable...
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.285      0.275  11.960    0.000
## log(Area)        0.402      0.043   9.443    0.000
## log(Nearest)    -0.041      0.118  -0.351    0.728
## sqrt(Scruz)     -0.049      0.045  -1.085    0.288
## log(Adjacent)   -0.024      0.046  -0.529    0.602
```

We will remove all the parameters that appear to be superfluous, and perform
an F-test to confirm that the simple model is sufficient.

```
m.s <- lm(log(Species) ~ log(Area), data=gala)
anova(m.s, m.c)
```

```
## Analysis of Variance Table
##
## Model 1: log(Species) ~ log(Area)
## Model 2: log(Species) ~ log(Area) + log(Nearest) + sqrt(Scruz) + log(Adjacent)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     28 17.218
## 2     25 15.299  3    1.9196 1.0456 0.3897
```

Next we will look at the coefficients.

```
summary(m.s)
```

```
##
## Call:
## lm(formula = log(Species) ~ log(Area), data = gala)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5442 -0.4001  0.0941  0.5449  1.3752
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.9037     0.1571  18.484  < 2e-16 ***
```

```
## log(Area)      0.3886      0.0416   9.342 4.23e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7842 on 28 degrees of freedom
## Multiple R-squared:  0.7571, Adjusted R-squared:  0.7484
## F-statistic: 87.27 on 1 and 28 DF,  p-value: 4.23e-10
```

The slope coefficient (0.3886) is the increase in log(Species) for every 1 unit increase in log(Area). Unfortunately that is not particularly convenient to interpretation and we will address this in the next section of this chapter.

Finally, we might be interested in creating a confidence interval for the expected number of tortoise species for an island with **Area=50**.

```
x0 <- data.frame(Area=50)
log.Species.CI <- predict(m.s, newdata=x0, interval='confidence')
log.Species.CI        # Log(Species) scale
```

```
##        fit      lwr      upr
## 1 4.423903 4.068412 4.779394
```

```
exp(log.Species.CI)  # Species scale
```

```
##        fit      lwr      upr
## 1 83.42122 58.46403 119.0322
```

Notice that on the species-scale, we see that the fitted value is not in the center of the confidence interval.

To help us understand what the log transformations are doing, we can produce a plot with the island Area on the x-axis and the expected number of Species on the y-axis and hopefully that will help us understand the relationship between the two.

```
library(ggplot2)
pred.data <- data.frame(Area=1:50)
pred.data <- pred.data %>%
  cbind( predict(m.s, newdata=pred.data, interval='conf'))
ggplot(pred.data, aes(x=Area)) +
  geom_line(aes(y=exp(fit))) +
  geom_ribbon(aes(ymin=exp(lwr), ymax=exp(upr)), alpha=.2) +
  ylab('Number of Species')
```
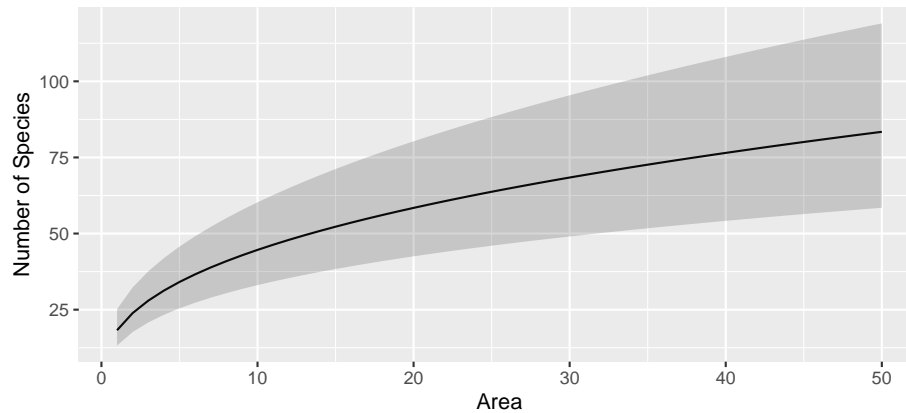
8.4   Interpretation of $\log$ transformed variable coefficients

One of the most difficult issues surrounding transformed variables is that the interpretation is difficult. Compared to taking the square root, log transformations are surprisingly interpretable on the original scale. Here we look at the interpretation of log transformed variables.

To investigate the effects of a log transformation, we'll examine a dataset that predicts the writing scores of $n = 200$ students using the gender, reading and math scores. This example was taken from the UCLA Statistical Consulting Group.

```r
file <- 'https://stats.idre.ucla.edu/wp-content/uploads/2016/02/lgtrans.csv'   # on the
file <- 'data-raw/lgtrans.csv'                                                  # on my
scores <- read.csv(file=file)
scores <- scores %>% rename(gender = female)

scores %>%
  dplyr::select(write, read, math, gender) %>%
  GGally::ggpairs( aes(color=gender),
    upper=list(continuous='points'), lower=list(continuous='cor'))
```

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

These data look pretty decent, and I'm not certain that I would do *any* transformation, but for the sake of having a concrete example that has both continuous and categorical covariates, we will interpret effects on a students' writing score.

8.4.1 Log-transformed response, un-transformed covariates

We consider the model where we have transformed the response variable and just an intercept term.

$$\log y = \beta_0 + \epsilon$$

```
model <- lm(log(write) ~ 1, data=scores)
broom::tidy(model)
```

```
## # A tibble: 1 x 5
##   term         estimate std.error statistic   p.value
##   <chr>           <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)      3.95    0.0137      288. 7.01e-263
```

We interpret the intercept as the mean of the log-transformed response values. We could back transform this to the original scale $\hat{y} = e^{\hat{\beta}_0} = e^{3.94835} = 51.85$ as a typical value of write. To distinguish this from the usually defined mean of the write values, we will call this as the *geometric mean*. Instead of calculating this by hand, we can have `emmeans()` do it for us.

```
emmeans(model, ~1)                          # Return y-hat value on the log-scale
```

```
## 1        emmean     SE  df lower.CL upper.CL
## overall    3.95 0.0137 199     3.92     3.98
```

```
##
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
```

```
emmeans(model, ~1, type='response')  # Return y-hat value on the original scale
```

```
##  1        response    SE  df lower.CL upper.CL
##  overall      51.8 0.71 199     50.5     53.3
##
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

Next we examine how to interpret the model when a categorical variable is added to the model.

$$\log y = \begin{cases} \beta_0 + \epsilon & \text{if female} \\ \beta_0 + \beta_1 + \epsilon & \text{if male} \end{cases}$$

```
model <- lm(log(write) ~ gender, data=scores)
broom::tidy(model)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    4.00     0.0179    223.   1.02e-239
## 2 gendermale    -0.103    0.0266     -3.89 1.39e-  4
```

The intercept is now the mean of the log-transformed `write` responses for the females and thus $e^{\hat{\beta}_0} = \hat{y}_f$ and the offset for males is the change in `log(write)` from the female group. Notice that for the males, we have

$$\log \hat{y}_m = \hat{\beta}_0 + \hat{\beta}_1$$
$$\hat{y}_m = e^{\hat{\beta}_0 + \hat{\beta}_1}$$
$$= \underbrace{e^{\hat{\beta}_0}}_{\hat{y}_f} * \underbrace{e^{\hat{\beta}_1}}_{\text{multiplier for males}}$$

and therefore we see that males tend to have writing scores $e^{-0.103} = 0.90 = 90\%$ of the females. Typically this sort of result would be reported as the males have a 10% lower writing score than the females.

Hand calculating these is challenging to do it correctly, but as usual we can have **emmeans** calculate it for us.

```
# I used reverse pairwise to get the ratio as male/female instead of female/male
emmeans(model, revpairwise~gender, type='response') %>%
  .[['contrasts']]
```

```
##  contrast      ratio    SE  df t.ratio p.value
##  male / female 0.902 0.024 198  -3.887  0.0001
##
## Tests are performed on the log scale
```

The model with a continuous covariate has a similar interpretation.

$$\log y = \begin{cases} \beta_0 + \beta_2 x + \epsilon & \text{if female} \\ \beta_0 + \beta_1 + \beta_2 x + \epsilon & \text{if male} \end{cases}$$

We will use the reading score read to predict the writing score. Then $\hat{\beta}_2$ is the predicted increase in `log(write)` for every 1-unit increase in read score. The interpretation of $\hat{\beta}_0$ is now $\log \hat{y}$ when $x = 0$ and therefore $\hat{y} = e^{\hat{\beta}_0}$ when $x = 0$.

```
model <- lm(log(write) ~ gender + read, data=scores)  # main effects model
broom::tidy(model)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    3.41     0.0546      62.5 8.68e-132
## 2 gendermale    -0.116    0.0210      -5.52 1.08e-  7
## 3 read           0.0113   0.00102     11.1  2.02e- 22
```

For females, we consider the difference in $\log \hat{y}$ for a 1-unit increase in x and will interpret this on the original write scale.

$$\log \hat{y}_f = \hat{\beta}_0 + \hat{\beta}_2 x$$
$$\hat{y}_f = e^{\hat{\beta}_0 + \hat{\beta}_2 x}$$

therefore we consider $e^{\hat{\beta}_2}$ as the multiplicative increase in write score for a 1-unit increase in x because of the following. Consider x_1 and $x_2 = x_1 + 1$. Then we consider the ratio of predicted values:

$$\frac{\hat{y}_2}{\hat{y}_1} = \frac{e^{\hat{\beta}_0 + \hat{\beta}_2\,(x+1)}}{e^{\hat{\beta}_0 + \hat{\beta}_2\,x}} = \frac{e^{\hat{\beta}_0} e^{\hat{\beta}_2\,x} e^{\hat{\beta}_2}}{e^{\hat{\beta}_0} e^{\hat{\beta}_2\,x}} = e^{\hat{\beta}_2}$$

For our writing scores example we have that $e^{\hat{\beta}_2} = e^{0.0113} = 1.011$ meaning there is an estimated 1% increase in `write` score for every 1-point increase in `read` score.

If we are interested in, say, a 20-unit increase in x, then that would result in an increase of

$$\frac{e^{\hat{\beta}_0 + \hat{\beta}_2\,(x+20)}}{e^{\hat{\beta}_0 + \hat{\beta}_2\,x}} = \frac{e^{\hat{\beta}_0} e^{\hat{\beta}_2\,x} e^{20\hat{\beta}_2}}{e^{\hat{\beta}_0} e^{\hat{\beta}_2\,x}} = e^{20\hat{\beta}_2} = \left(e^{\hat{\beta}_2}\right)^{20}$$

and for the writing scores we have

$$e^{20\hat{\beta}_2} = \left(e^{\hat{\beta}_2}\right)^{20} = 1.0113^{20} = 1.25$$

or a 22% increase in writing score for a 20-point increase in reading score.

```
# to make emmeans calculate this, we must specify a 1-unit or 20-unit increase
emmeans(model, pairwise ~ read, at=list(read=c(51,50)), type='response') %>%
  .[['contrasts']]
```

```
##  contrast ratio       SE  df t.ratio p.value
##  51 / 50  1.011 0.001032 197 11.057  <.0001
##
## Results are averaged over the levels of: gender
## Tests are performed on the log scale
```

```
emmeans(model, pairwise ~ read, at=list(read=c(90,70)), type='response') %>%
  .[['contrasts']]
```

```
##  contrast ratio      SE  df t.ratio p.value
##  90 / 70   1.25 0.0256 197 11.057  <.0001
##
## Results are averaged over the levels of: gender
## Tests are performed on the log scale
```

In short, we can interpret $e^{\hat{\beta}_i}$ as the multiplicative increase/decrease in the non-transformed response variable. Some students get confused by what is meant by a % increase or decrease in y.

- A 75% decrease in y has a resulting value of $(1 - 0.75)\,y = (0.25)\,y$
- A 75% increase in y has a resulting value of $(1 + 0.75)\,y = (1.75)\,y$
- A 100% increase in y has a resulting value of $(1 + 1.00)\,y = 2y$ and is a doubling of y.
- A 50% decrease in y has a resulting value of $(1 - 0.5)\,y = (0.5)\,y$ and is a halving of y.

8.4.2   Un-transformed response, log-transformed covariate

We consider the model

$$y = \beta_0 + \beta_2 \log x + \epsilon$$

and consider two different values of x (which we'll call x_1 and x_2 and we are considering the effect of moving from x_1 to x_2) and look at the differences between the predicted values $\hat{y}_2 - \hat{y}_1$.

$$\begin{aligned}
\hat{y}_2 - \hat{y}_1 &= \left[\hat{\beta}_0 + \hat{\beta}_2 \log x_2\right] - \left[\hat{\beta}_0 + \hat{\beta}_2 \log x_1\right] \\
&= \hat{\beta}_2 \left[\log x_2 - \log x_1\right] \\
&= \hat{\beta}_2 \log \left[\frac{x_2}{x_1}\right]
\end{aligned}$$

This means that so long as the ratio between the two x-values is constant, then the change in $\hat{y}$ is the same. So doubling the value of x from 1 to 2 has the same effect on $\hat{y}$ as changing x from 50 to 100.

```
model <- lm( write ~ gender + log(read), data=scores)
broom::tidy(model)
```

```
## # A tibble: 3 x 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     -59.1      9.95     -5.94 1.28e- 8
## 2 gendermale       -5.43     1.01     -5.36 2.29e- 7
## 3 log(read)        29.0      2.53     11.5  9.98e-24
```

```
# predict writing scores for three females,
# each with a reading score 50% larger than the other previous
predict(model, newdata=data.frame(gender=rep('female',3),
                                   read=c(40, 60, 90)))
```

```
##        1        2        3
## 48.06622 59.84279 71.61936
```

We should see a

$$29.045 \, \log \left(1.5\right) = 11.78$$

difference in $\hat{y}$ values for the first and second students and the second and third.

```
emmeans(model, revpairwise~log(read), at=list(read=c(2,4,8)))
```

```
## $emmeans
##  read emmean   SE  df lower.CL upper.CL
##     2 -41.66 8.21 197    -57.9   -25.46
##     4 -21.53 6.47 197    -34.3    -8.78
##     8  -1.39 4.72 197    -10.7     7.92
##
## Results are averaged over the levels of: gender
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate   SE  df t.ratio p.value
##  4 - 2         20.1 1.75 197 11.493  <.0001
##  8 - 2         40.3 3.50 197 11.493  <.0001
##  8 - 4         20.1 1.75 197 11.493  <.0001
##
## Results are averaged over the levels of: gender
## P value adjustment: tukey method for comparing a family of 3 estimates
```

8.4.3   Log-transformed response, log-transformed covariate

This combines the interpretations in the previous two sections. We consider

$$\log y = \beta_0 + \beta_2 \log x + \epsilon$$

and we again consider two x values (again x_1 and x_2). We then examine the difference in the $\log \hat{y}$ values as

$$\log \hat{y}_2 - \log \hat{y}_1 = \left[\hat{\beta}_0 + \hat{\beta}_2 \log x_2\right] - \left[\hat{\beta}_0 + \hat{\beta}_2 \log x_1\right]$$

$$\log\left[\frac{\hat{y}_2}{\hat{y}_1}\right] = \hat{\beta}_2 \log\left[\frac{x_2}{x_1}\right]$$

$$\log\left[\frac{\hat{y}_2}{\hat{y}_1}\right] = \log\left[\left(\frac{x_2}{x_1}\right)^{\hat{\beta}_2}\right]$$

$$\frac{\hat{y}_2}{\hat{y}_1} = \left(\frac{x_2}{x_1}\right)^{\hat{\beta}_2}$$

This allows us to examine the effect of some arbitrary percentage increase in x value as a percentage increase in y value.

```
model <- lm(log(write) ~ gender + log(read), data=scores)
broom::tidy(model)
```

```
## # A tibble: 3 x 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)      1.71     0.205      8.36 1.14e-14
## 2 gendermale     -0.114    0.0209     -5.48 1.27e- 7
## 3 log(read)       0.581    0.0521     11.1  1.08e-22
```

which implies for a 10% increase in `read` score, we should see a $1.10^{0.581} = 1.056$ multiplier in `write` score. That is to say, a 10% increase in reading score results in a 5% increase in writing score.

```
emmeans(model, pairwise~log(read), at=list(read=c(55,50)),
        var='log(read)', type='response')
```

```
## $emmeans
##  read response    SE  df lower.CL upper.CL
##    55     53.8 0.594 197     52.6     54.9
##    50     50.9 0.535 197     49.8     51.9
##
## Results are averaged over the levels of: gender
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
##
## $contrasts
##  contrast ratio      SE  df t.ratio p.value
##  55 / 50   1.06 0.00525 197 11.148   <.0001
##
## Results are averaged over the levels of: gender
## Tests are performed on the log scale
```

For the Galapagos islands, we had

```
m.s <- lm(log(Species) ~ log(Area), data=gala)
broom::tidy(m.s)
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)      2.90     0.157      18.5  3.17e-17
## 2 log(Area)        0.389    0.0416      9.34 4.23e-10
```

```r
emmeans(m.s, pairwise~Area, at=list(Area=c(400, 200)), type='response')
```

```
## $emmeans
##  Area response   SE df lower.CL upper.CL
##   400       187 43.7 28    116.0      302
##   200       143 30.2 28     92.7      221
##
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
##
## $contrasts
##  contrast  ratio     SE df t.ratio p.value
##  400 / 200  1.31 0.0377 28 9.342   <.0001
##
## Tests are performed on the log scale
```

and therefore doubling of Area (i.e. the ratio of the $Area_2/Area_1 = 2$) results in a $2^{0.389} = 1.31$ multiplier of the `Species` value. That is to say doubling the island area increases the number of species by 31%.

In the table below β represents the group offset value, or the slope value associated with x. If we are in a model with multiple slopes such as an ANCOVA model, then the beta term represents the slope of whatever group you are interested.

Response	Explanatory	Term	Interpretation
$\log(y)$	Categorical	e^β	Switching from the reference group results in this *multiplicative* change on y.
$\log(y)$	Continuous x	e^β	A 1-unit change in x results in this *multiplicative* change on y.
$\log(y)$	Continuous x	$\left(e^\beta\right)^\delta$	A δ-unit change in x results in this *multiplicative* change on y.
y	Continuous $\log(x)$	$\beta \log\left(\frac{x_2}{x_1}\right)$	The proportional change in x results in an *additive* change on y.

Response	Explanatory	Term	Interpretation
$\log(y)$	Continuous $\log(x)$	$\left(\frac{x_2}{x_1}\right)^{\beta}$	The proportional change in x results in the *multiplicative* change on y.

8.5 Exercises

1. In the ANCOVA chapter, we examined the relationship on dose of vitamin C on guinea pig tooth growth based on how the vitamin was delivered (orange juice vs a pill supplement).

   a. Load the `ToothGrowth` data which is available in base R.
   b. Plot the data with log dose level on the x-axis and tooth length growth on the y-axis. Color the points by supplement type.
   c. Fit a linear model using the log transformed dose.
   d. Interpret the effect of doubling the dose on tooth growth for the OJ and VC supplement groups.

2. We will consider the relationship between income and race using a subset of employed individuals from the American Community Survey.

   a. Load the `EmployedACS` dataset from the `Lock5Data` package.
   b. Create a box plot showing the relationship between `Race` and `Income`.
   c. Consider the boxcox family of transformations of `Income`. What transformation seems appropriate? Consider both square-root and log transformation? While the race differences are not statistically significant in either case, there is an interesting shift in how black, white, and other groups are related. *Because there are people with zero income, we have to do something. We could either use a transformation like $\sqrt{y}$, remove all the zero observations, or to add a small value to the zero observations. We'll add 0.05 to the zero values, which represents the zero income people receiving* $50*. Graph both the log and square root transformations. Do either completely address the issue? What about the cube-root ($\lambda = 1/3$)?*
   d. Using your cube-root transformed `Income` variable, fit an ANOVA model and evaluate the relationship between race and income utilizing these data. Provide (and interpret) the point estimates, even though they aren't statistically significant. *Importantly, we haven't accounted for many sources of variability such as education level and job type. There is much more to consider than just this simple analysis.*

3. The dataset `Lock5Data::HomesForSale` has a random sample of home prices in 4 different states. Consider a regression model predicting the

`Price` as a function of `Size`, `Bedrooms`, and `Baths`.

a) Examine a scatterplot of `Price` and `Size` and justify a log transformation to both.
b) Build an appropriate model considering only main effects. Find any observations that are unusual and note any decisions about including correlated variables.
c) Calculate and interpret the estimated effect of a 10% increase in home size on the price.
d) Calculate and interpret the difference between California and Pennsylvania in terms of average house price.

Correlated Covariates

```r
library(tidyverse)
```

Interpretation with Correlated Covariates

The standard interpretation of the slope parameter is that β_j is the amount of increase in y for a one unit increase in the jth covariate, provided that all other covariates stayed the same.

The difficulty with this interpretation is that covariates are often related, and the phrase "all other covariates stayed the same" is often not reasonable. For example, if we have a dataset that models the mean annual temperature of a location as a function of latitude, longitude, and elevation, then it is not physically possible to hold latitude, and longitude constant while changing elevation.

One common issue that make interpretation difficult is that covariates can be highly correlated.

Perch Example: We might be interested in estimating the weight of a fish based off of its length and width. The dataset we will consider is from fishes are caught from the same lake (Laengelmavesi) near Tampere in Finland. The following variables were observed:

Variable	Interpretation
Weight	Weight (g)
Length.1	Length from nose to beginning of Tail (cm)
Length.2	Length from nose to notch of Tail (cm)
Length.3	Length from nose to tip of tail (cm)
Height	Maximal height as a percentage of Length.3
Width	Maximal width as a percentage of Length.3
Sex	0=Female, 1=Male
Species	Which species of perch (1-7)

We first look at the data and observe the expected relationship between length
and weight.

```r
# File location on the internet or on my local computer.
file <- 'https://raw.githubusercontent.com/dereksonderegger/571/master/data-raw/Fish.c
file <- '~/github/571/data-raw/Fish.csv'

fish <- read.table(file, header=TRUE, skip=111, sep=',') %>%
  filter( !is.na(Weight) )

### generate a pairs plot in a couple of different ways...
# pairs(fish)
# pairs( Weight ~ Length.1 + Length.2 + Length.3 + Height + Width, data=fish )
# pairs( Weight ~ ., data=fish )

fish %>%
  dplyr::select(Weight, Length.1, Length.2, Length.3, Height, Width) %>%
  GGally::ggpairs(upper=list(continuous='points'),
                  lower=list(continuous='cor'))
```
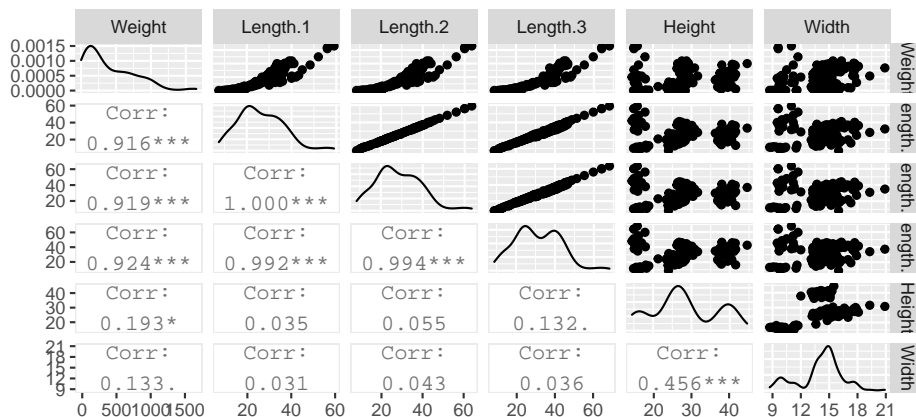

Naively, we might consider the linear model with all the length effects present.

```r
model <- lm(Weight ~ Length.1 + Length.2 + Length.3 + Height + Width, data=fish)
summary(model)
```

```
##
## Call:
## lm(formula = Weight ~ Length.1 + Length.2 + Length.3 + Height +
##     Width, data = fish)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -302.22  -79.72  -39.88    92.63   344.85
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -724.539      77.133  -9.393   <2e-16 ***
## Length.1      32.389      45.134   0.718   0.4741
## Length.2      -9.184      48.367  -0.190   0.8497
## Length.3       8.747      16.283   0.537   0.5919
## Height         4.947       2.768   1.787   0.0759 .
## Width          8.636       6.972   1.239   0.2174
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 132.9 on 152 degrees of freedom
## Multiple R-squared:  0.8675, Adjusted R-squared:  0.8631
## F-statistic:    199 on 5 and 152 DF,  p-value: < 2.2e-16
```

This is crazy. There is a negative relationship between `Length.2` and `Weight`. That does not make any sense unless you realize that this is the effect of `Length.2` assuming the other covariates are in the model and can be held constant while changing the value of `Length.2`, which is obviously ridiculous.

If we remove the highly correlated covariates then we see a much better behaved model

```
model <- lm(Weight ~ Length.2 + Height + Width, data=fish)
summary(model)
```

```
##
## Call:
## lm(formula = Weight ~ Length.2 + Height + Width, data = fish)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -306.14  -75.11  -36.45   89.54  337.95
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -701.0750     71.0438  -9.868  < 2e-16 ***
## Length.2      30.4360      0.9841  30.926  < 2e-16 ***
## Height         5.5141      1.4311   3.853 0.000171 ***
## Width          5.6513      5.2016   1.086 0.278974
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 132.3 on 154 degrees of freedom
## Multiple R-squared:  0.8669, Adjusted R-squared:  0.8643
## F-statistic: 334.2 on 3 and 154 DF,  p-value: < 2.2e-16
```

When you have two variables in a model that are highly positively correlated, you often find that one will have a positive coefficient and the other will be negative. Likewise, if two variables are highly negatively correlated, the two regression coefficients will often be the same sign.

In this case the sum of the three length covariate estimates was approximately 31 in both cases, but with three length variables, the second could be negative the third be positive with approximately the same magnitude and we get approximately the same model as with both the second and third length variables missing from the model.

$$y_i = \beta_0 + \beta_1 L_1 + \beta_2 L_2 + \beta_3 L_3 + \epsilon_i$$

but if the covariates are highly correlated, then approximately $L_1 = L_2 = L_3 = L$ and this equation could be written:

$$y_i = \beta_0 + (\beta_1 + \beta_2 + \beta_3)L_3 + \epsilon_i$$

$$y_i = \beta_0 + [(\beta_1 - 10) + \beta_2 + (\beta_3 + 10)] L_3 + \epsilon_i$$

In general, you should be very careful with the interpretation of the regression coefficients when the covariates are highly correlated.

Solutions

In general you have three ways to deal with correlated covariates.

1. Select one of the correlated covariates to include in the model. You could either select the covariate with the highest correlation with the response. Alternatively you could use your scientific judgment as to which covariate is most appropriate.

2. You could create an *index* variable that combines the correlated covariates into a single amalgamation variable. For example, we could create $L = (L_1 + L_2 + L_3)/3$ to be the average of the three length measurements.

3. We could use some multivariate method such as *Principle Components Analysis* to inform us on how to make the index variable. We won't address this in this here, though.

4. Just not worry about it. Remember, the only problem is with the *interpretation* of the parameters. If we only care about making accurate predictions within the scope of our data, then including highly correlated variables doesn't matter. However, if you are making predictions away from the data, the flipped signs might be a big deal.

Chapter 9

Variable Selection

```r
library(tidyverse)    # dplyr, tidyr, ggplot2, etc
```

Given a set of data, we are interested in selecting the best subset of predictors for the following reasons:

1. Occam's Razor tells us that from a list of plausible model or explanations, the simplest is usually the best. In the statistics sense, I want the smallest model that adequately explains the observed data patterns.

2. Unnecessary predictors add noise to the estimates of other quantities and will waste degrees of freedom, possibly increasing the estimate of $\hat{\sigma}^2$.

3. We might have variables that are co-linear.

The problems that arise in the diagnostics of a model will often lead a researcher to consider other models, for example to include a quadratic term to account for curvature. The model building process is often an iterative procedure where we build a model, examine the diagnostic plots and consider what could be added or modified to correct issues observed.

9.1 Nested Models

Often one model is just a simplification of another and can be obtained by setting some subset of β_i values equal to zero. Those models can be adequately compared by the F-test, which we have already made great use of.

We should be careful to note that we typically do not want to remove the main covariate from the model if the model uses the covariate in a more complicated fashion. For example, if my model is

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$$

where $\epsilon \sim N\left(0, \sigma^2\right)$, then considering the simplification $\beta_1 = 0$ and removing the effect of x is not desirable because that forces the parabola to be symmetric about $x = 0$. Similarly, if the model contains an interaction effect, then the removal of the main effect drastically alters the interpretation of the interaction coefficients and should be avoided. Often times removing a lower complexity term while keeping a higher complexity term results in unintended consequences and is typically not recommended.

9.2   Testing-Based Model Selection

Starting with a model that is likely too complex, consider a list of possible terms to remove and remove each in turn while evaluating the resulting model to the starting model using an F-test. Whichever term has the highest p-value is removed and the process is repeated until no more terms have non-significant p-values. This is often referred to as *backward selection*.

It should be noted that the cutoff value for significance here does not have to be $\alpha = 0.05$. If prediction performance is the primary goal, then a more liberal α level is appropriate.

Starting with a model that is likely too small, consider adding terms until there are no more terms that when added to the model are significant. This is called *forward selection*.

This is a hybrid between forward selection and backward elimination. At every stage, a term is either added or removed. This is referred to as *stepwise selection*.

Stepwise, forward, and backward selection are commonly used but there are some issues.

1. Because of the "one-at-a-time" nature of the addition/deletion, the most optimal model might not be found.

2. p-values should not be treated literally. Because the multiple comparisons issue is completely ignored, the p-values are lower than they should be if multiple comparisons were accounted for. As such, it is possible to sort through a huge number of potential covariates and find one with a low p-value simply by random chance. This is "data dredging" and is a serious issue.

3. As a non-thinking algorithm, these methods ignore the science behind that data and might include two variables that are highly collinear or might ignore variables that are scientifically interesting.

9.2.1 Example - U.S. Life Expectancy

Using data from the Census Bureau we can look at the life expectancy as a response to a number of predictors. One R function that is often convenient to use is the update() function that takes a `lm()` object and adds or removes things from the formula. The notation . ~ . means to leave the response and all the predictors alone, while . ~ . + `vnew` will add the main effect of `vnew` to the model.

```r
data('state')  # loads a matrix state.x77 and a vector of stat abbreviations

# Convert from a matrix to a data frame with state abbreviations
state.data <- data.frame(state.x77, row.names=state.abb)
str(state.data)
```

```
## 'data.frame':    50 obs. of  8 variables:
##  $ Population: num  3615 365 2212 2110 21198 ...
##  $ Income    : num  3624 6315 4530 3378 5114 ...
##  $ Illiteracy: num  2.1 1.5 1.8 1.9 1.1 0.7 1.1 0.9 1.3 2 ...
##  $ Life.Exp  : num  69 69.3 70.5 70.7 71.7 ...
##  $ Murder    : num  15.1 11.3 7.8 10.1 10.3 6.8 3.1 6.2 10.7 13.9 ...
##  $ HS.Grad   : num  41.3 66.7 58.1 39.9 62.6 63.9 56 54.6 52.6 40.6 ...
##  $ Frost     : num  20 152 15 65 20 166 139 103 11 60 ...
##  $ Area      : num  50708 566432 113417 51945 156361 ...
```

We should first look at the

```r
state.data %>%
  dplyr::select( Life.Exp, Population:Area ) %>%
  GGally::ggpairs( upper=list(continuous='points'), lower=list(continuous='cor') )
```
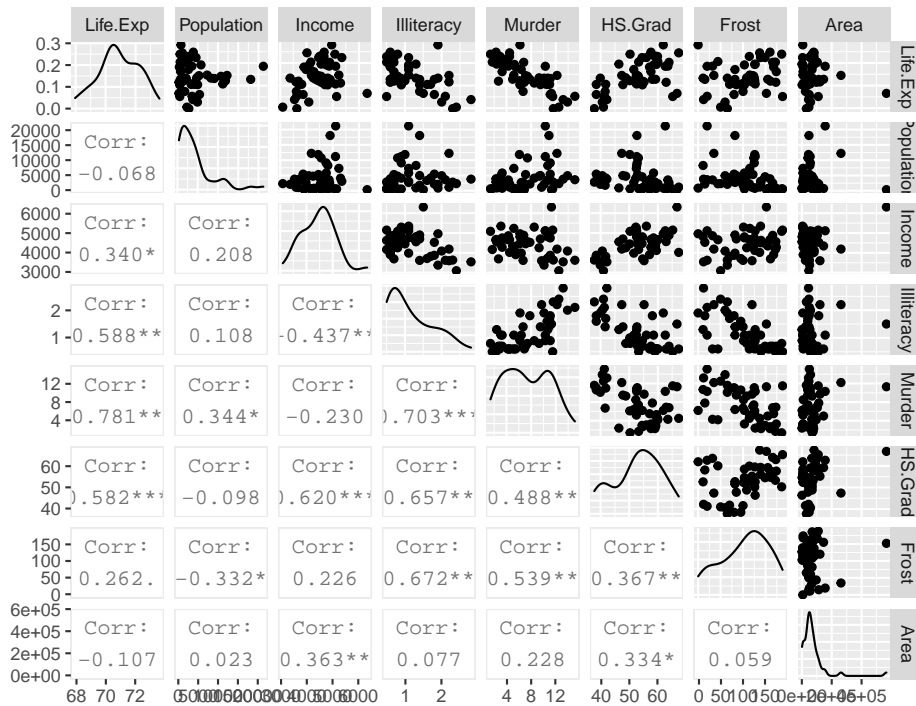
I want to add a quadratic effect for `HS.Grad` rate and for `Income`. Also, we see that `Population` and `Area` seem to have some high skew to their distributions, so a log transformation might help. We'll modify the data and then perform the backward elimination method starting with the model with all predictors as main effects.

```
state.data <- state.data %>%
  mutate( HS.Grad.2 = HS.Grad ^ 2,
          Income.2  = Income  ^ 2,
          Log.Population = log(Population),
          Log.Area = log(Area)) %>%
  dplyr::select( -Population, -Area )   # remove the original Population and Area cova

# explicitly define my starting model
m1 <- lm(Life.Exp ~ Log.Population + Income + Illiteracy +
                    Murder + HS.Grad + Frost + HS.Grad.2 + Income.2 +  Log.Area, data=s
#
# Define the same model, but using shorthand
# The '.' means everything else in the data frame
m1 <- lm( Life.Exp ~ ., data=state.data)
summary(m1)$coefficients %>% round(digits=3)
```

```
##                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)          61.174       6.984    8.759      0.000
## Income                0.004       0.003    1.517      0.137
## Illiteracy            0.367       0.388    0.946      0.350
## Murder               -0.304       0.049   -6.214      0.000
## HS.Grad              -0.031       0.208   -0.151      0.881
## Frost                -0.004       0.003   -1.068      0.292
## HS.Grad.2             0.001       0.002    0.394      0.696
## Income.2              0.000       0.000   -1.518      0.137
## Log.Population        0.191       0.150    1.273      0.211
## Log.Area              0.100       0.113    0.885      0.382
```

The signs make reasonable sense (higher murder rates decrease life expectancy) but covariates like `Income` are not significant, which is surprising. The largest p-value is `HS.Grad`. However, I don't want to remove the lower-order graduation term and keep the squared-term. So instead I will remove both of them since they are the highest p-values. Notice that `HS.Grad` is correlated with `Income` and `Illiteracy`.

```
# Remove Graduation Rate from the model from the model
m1 <- update(m1, .~. - HS.Grad - HS.Grad.2)
summary(m1)$coefficients %>% round(digits=3)
```

```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        60.894      6.100   9.983    0.000
## Income              0.004      0.002   1.711    0.094
## Illiteracy          0.087      0.373   0.233    0.817
## Murder             -0.318      0.048  -6.686    0.000
## Frost              -0.006      0.003  -1.807    0.078
## Income.2            0.000      0.000  -1.581    0.121
## Log.Population      0.041      0.132   0.309    0.759
## Log.Area            0.206      0.103   1.995    0.053
```

```
# Next remove Illiteracy
m1 <- update(m1, .~. - Illiteracy)
summary(m1)$coefficients %>% round(digits=3)
```

```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        61.779      4.717  13.098    0.000
## Income              0.004      0.002   1.872    0.068
## Murder             -0.314      0.042  -7.423    0.000
## Frost              -0.006      0.003  -2.345    0.024
## Income.2            0.000      0.000  -1.699    0.097
## Log.Population      0.041      0.130   0.314    0.755
## Log.Area            0.198      0.097   2.048    0.047
```

```r
# And Log.Population...
m1 <- update(m1, .~. - Log.Population)
summary(m1)$coefficients %>% round(digits=3)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   61.413      4.523  13.577    0.000
## Income         0.004      0.002   2.257    0.029
## Murder        -0.309      0.039  -7.828    0.000
## Frost         -0.006      0.002  -2.612    0.012
## Income.2       0.000      0.000  -2.044    0.047
## Log.Area       0.200      0.096   2.091    0.042
```

The removal of `Income.2` is a tough decision because the p-value is very close to $\alpha = 0.05$ and might be left in if it makes model interpretation easier or if the researcher feels a quadratic effect in income is appropriate (perhaps rich people are too stressed?).

```r
summary(m1)
```

```
##
## Call:
## lm(formula = Life.Exp ~ Income + Murder + Frost + Income.2 +
##     Log.Area, data = state.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.28858 -0.50631 -0.07242  0.49738  1.75839
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.141e+01  4.523e+00  13.577  < 2e-16 ***
## Income       4.212e-03  1.867e-03   2.257   0.0290 *
## Murder      -3.092e-01  3.950e-02  -7.828 7.14e-10 ***
## Frost       -6.487e-03  2.483e-03  -2.612   0.0123 *
## Income.2    -4.188e-07  2.049e-07  -2.044   0.0470 *
## Log.Area     2.002e-01  9.576e-02   2.091   0.0424 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7349 on 44 degrees of freedom
## Multiple R-squared:  0.7309, Adjusted R-squared:  0.7003
## F-statistic:  23.9 on 5 and 44 DF,  p-value: 1.549e-11
```

We are left with a model that adequately explains `Life.Exp` but we should be careful to note that just because a covariate was removed from the model

does not imply that it isn't related to the response. For example, being a high school graduate is highly correlated with not being illiterate as is `Income` and thus replacing `Illiteracy` shows that illiteracy is associated with lower life expectancy, but it is not as predictive as `Income`.

```r
m2 <- lm(Life.Exp ~ Illiteracy+Murder+Frost, data=state.data)
summary(m2)
```

```
##
## Call:
## lm(formula = Life.Exp ~ Illiteracy + Murder + Frost, data = state.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.59010 -0.46961  0.00394  0.57060  1.92292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 74.556717   0.584251 127.611  < 2e-16 ***
## Illiteracy  -0.601761   0.298927  -2.013  0.04998 *
## Murder      -0.280047   0.043394  -6.454 6.03e-08 ***
## Frost       -0.008691   0.002959  -2.937  0.00517 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7911 on 46 degrees of freedom
## Multiple R-squared:  0.6739, Adjusted R-squared:  0.6527
## F-statistic: 31.69 on 3 and 46 DF,  p-value: 2.915e-11
```

Notice that the R^2 values for both models are quite similar 0.7309 vs 0.6739 but the first model with the higher R^2 has one more predictor variable? Which model should I prefer? I can't do an F-test because these models are not nested.

9.3 Criterion Based Procedures

9.3.1 Information Criterions

It is often necessary to compare models that are not nested. For example, I might want to compare

$$y = \beta_0 + \beta_1 x + \epsilon$$

vs

$$y = \beta_0 + \beta_2 w + \epsilon$$

This comparison comes about naturally when doing forward model selection and we are looking for the "best" covariate to add to the model first.

Akaike introduced his criterion (which he called "An Information Criterion") as

$$AIC = \underbrace{-2 \log L\left(\hat{\beta}, \hat{\sigma}|\,\text{data}\right)}_{\text{decreases if RSS decreases}} + \underbrace{2p}_{\text{increases as p increases}}$$

where $L\left(\hat{\beta}|\,\text{data}\right)$ is the likelihood function and p is the number of elements in the $\hat{\beta}$ vector and we regard a lower AIC value as better. Notice the $2p$ term is essentially a penalty on adding addition covariates so to lower the AIC value, a new predictor must lower the negative log likelihood more than it increases the penalty.

To convince ourselves that the first summand decreases with decreasing RSS in the standard linear model, we examine the likelihood function

$$f\left(y\,|\,\beta, \sigma, X\right) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left[-\frac{1}{2\sigma^2}\left(y - X\beta\right)^T \left(y - X\beta\right)\right]$$
$$= L\left(\beta, \sigma\,|\,y, X\right)$$

and we could re-write this as

$$\log L\left(\hat{\beta}, \hat{\sigma}\,|\,\text{data}\right) = -\log\left((2\pi\hat{\sigma}^2)^{n/2}\right) - \frac{1}{2\hat{\sigma}^2}\left(y - X\hat{\beta}\right)^T\left(y - X\hat{\beta}\right)$$
$$= -\frac{n}{2}\log\left(2\pi\hat{\sigma}^2\right) - \frac{1}{2\hat{\sigma}^2}\left(y - X\hat{\beta}\right)^T\left(y - X\hat{\beta}\right)$$
$$= -\frac{1}{2}\left[n\log\left(2\pi\hat{\sigma}^2\right) + \frac{1}{\hat{\sigma}^2}\left(y - X\hat{\beta}\right)^T\left(y - X\hat{\beta}\right)\right]$$
$$= -\frac{1}{2}\left[+n\log\left(2\pi\right) + n\log\hat{\sigma}^2 + \frac{1}{\hat{\sigma}^2}RSS\right]$$

It isn't clear what we should do with the $n\log\left(2\pi\right)$ term in the $\log L()$ function. There are some compelling reasons to ignore it and just use the second, and there are reasons to use both terms. Unfortunately, statisticians have not settled on one convention or the other and different software packages might therefore report different values for AIC.

As a general rule of thumb, if the difference in AIC values is less than two then the models are not significantly different, differences between 2 and 4 AIC units are marginally significant and any difference greater than 4 AIC units is highly significant.

Notice that while this allows us to compare models that are not nested, it does require that the same data are used to fit both models. Because I could start out with my data frame including both x and x^2, (or more generally x and $f\left(x\right)$ for some function $f()$) you can regard a transformation of a covariate as "the

same data". However, a transformation of a y-variable is not and therefore we cannot use AIC to compare a models `log(y) ~ x` versus the model `y ~ x`.

Another criterion that might be used is *Bayes Information Criterion* (BIC) which is

$$BIC = -2 \log L\left(\hat{\beta}, \hat{\sigma} | \text{data}\right) + p \log n$$

and this criterion punishes large models more than AIC does (because $\log n > 2$ for $n \geq 8$)

The AIC value of a linear model can be found using the AIC() on a `lm()()` object.

```
AIC(m1)
```

```
## [1] 118.6942
```

```
AIC(m2)
```

```
## [1] 124.2947
```

Because the AIC value for the first model is lower, we would prefer the first model that includes both `Income` and `Income.2` compared to model 2, which was `Life.Exp ~ Illiteracy+Murder+Frost`.

9.3.2 Adjusted `R-sq`

One of the problems with R^2 is that it makes no adjustment for how many parameters in the model. Recall that R^2 was defined as

$$R^2 = \frac{RSS_S - RSS_C}{RSS_S} = 1 - \frac{RSS_C}{RSS_S}$$

where the simple model was the intercept only model. We can create an R^2_{adj} statistic that attempts to add a penalty for having too many parameters by defining

$$R^2_{adj} = 1 - \frac{RSS_C/(n-p)}{RSS_S/(n-1)}$$

With this adjusted definition, adding a variable to the model that has no predictive power will decrease R^2_{adj}.

9.3.3 Example

Returning to the life expectancy data, we could start with a simple model add covariates to the model that have the lowest AIC values. R makes this easy with the function `add1()` which will take a linear model (which includes the data frame that originally defined it) and will sequentially add all of the possible terms that are not currently in the model and report the AIC values for each model.

```
# Define the biggest model I wish to consider
biggest <- Life.Exp ~ Log.Population + Income + Illiteracy + Murder +
                      HS.Grad + Frost + Log.Area + HS.Grad.2 + Income.2

# Define the model I wish to start with
m <- lm(Life.Exp ~ 1, data=state.data)

add1(m, scope=biggest)  # what is the best addition to make?
```

```
## Single term additions
##
## Model:
## Life.Exp ~ 1
##                Df Sum of Sq    RSS     AIC
## <none>                       88.299  30.435
## Log.Population  1     1.054 87.245  31.835
## Income          1    10.223 78.076  26.283
## Illiteracy      1    30.578 57.721  11.179
## Murder          1    53.838 34.461 -14.609
## HS.Grad         1    29.931 58.368  11.737
## Frost           1     6.064 82.235  28.878
## Log.Area        1     1.042 87.257  31.842
## HS.Grad.2       1    27.414 60.885  13.848
## Income.2        1     7.464 80.835  28.020
```

Clearly the addition of `Murder` to the model results in the lowest AIC value, so we will add `Murder` to the model. Notice the `<none>` row corresponds to the model m which we started with and it has a `RSS=88.299`. For each model considered, R will calculate the `RSS_{C}` for the new model and will calculate the difference between the starting model and the more complicated model and display this in the Sum of Squares column.

```
m <- update(m, . ~ . + Murder)  # add murder to the model
add1(m, scope=biggest)          # what should I add next?
```

```
## Single term additions
```

```
##
## Model:
## Life.Exp ~ Murder
##                 Df Sum of Sq     RSS     AIC
## <none>                        34.461 -14.609
## Log.Population  1    2.9854 31.476 -17.140
## Income          1    2.4047 32.057 -16.226
## Illiteracy      1    0.2732 34.188 -13.007
## HS.Grad         1    4.6910 29.770 -19.925
## Frost           1    3.1346 31.327 -17.378
## Log.Area        1    1.4583 33.003 -14.771
## HS.Grad.2       1    4.4396 30.022 -19.505
## Income.2        1    1.8972 32.564 -15.441
```

There is a companion function to `add1()` that finds the best term to drop. It is conveniently named `drop1()` but here the `scope` parameter defines the smallest model to be considered.

It would be nice if all of this work was automated. Again, R makes our life easy and the function `step()` does exactly this. The set of models searched is determined by the scope argument which can be a *list* of two formulas with components upper and lower or it can be a single formula, or it can be blank. The right-hand-side of its lower component defines the smallest model to be considered and the right-hand-side of the upper component defines the largest model to be considered. If `scope` is a single formula, it specifies the upper component, and the lower model taken to be the intercept-only model. If scope is missing, the initial model is used as the upper model.

```r
smallest <- Life.Exp ~ 1
biggest <- Life.Exp ~ Log.Population + Income + Illiteracy +
                    Murder + HS.Grad + Frost + Log.Area + HS.Grad.2 + Income.2
m <- lm(Life.Exp ~ Income, data=state.data)
stats::step(m, scope=list(lower=smallest, upper=biggest))
```

```
## Start:  AIC=26.28
## Life.Exp ~ Income
##
##                   Df Sum of Sq    RSS     AIC
## + Murder           1    46.020 32.057 -16.226
## + Illiteracy       1    21.109 56.968  12.523
## + HS.Grad          1    19.770 58.306  13.684
## + Income.2         1    19.062 59.015  14.288
## + HS.Grad.2        1    17.193 60.884  15.847
## + Frost            1     3.188 74.889  26.199
## <none>                          78.076  26.283
## + Log.Population   1     1.298 76.779  27.445
```

```
## + Log.Area        1     0.994 77.082  27.642
## - Income          1    10.223 88.299  30.435
##
## Step:  AIC=-16.23
## Life.Exp ~ Income + Murder
##
##                  Df Sum of Sq    RSS     AIC
## + Frost           1     3.918 28.138 -20.745
## + Income.2        1     3.036 29.021 -19.200
## + HS.Grad         1     2.388 29.668 -18.097
## + Log.Population  1     2.371 29.686 -18.068
## + HS.Grad.2       1     2.199 29.857 -17.780
## <none>                         32.057 -16.226
## + Log.Area        1     1.229 30.827 -16.181
## - Income          1     2.405 34.461 -14.609
## + Illiteracy      1     0.011 32.046 -14.242
## - Murder          1    46.020 78.076  26.283
##
## Step:  AIC=-20.74
## Life.Exp ~ Income + Murder + Frost
##
##                  Df Sum of Sq    RSS     AIC
## + HS.Grad         1     2.949 25.189 -24.280
## + HS.Grad.2       1     2.764 25.375 -23.914
## + Log.Area        1     2.122 26.017 -22.664
## + Income.2        1     2.017 26.121 -22.465
## <none>                         28.138 -20.745
## + Illiteracy      1     0.950 27.189 -20.461
## + Log.Population  1     0.792 27.347 -20.172
## - Income          1     3.188 31.327 -17.378
## - Frost           1     3.918 32.057 -16.226
## - Murder          1    46.750 74.889  26.199
##
## Step:  AIC=-24.28
## Life.Exp ~ Income + Murder + Frost + HS.Grad
##
##                  Df Sum of Sq    RSS     AIC
## + Log.Population  1     2.279 22.911 -27.021
## + Income.2        1     1.864 23.326 -26.124
## - Income          1     0.182 25.372 -25.920
## <none>                         25.189 -24.280
## + Log.Area        1     0.570 24.619 -23.425
## + HS.Grad.2       1     0.218 24.972 -22.714
## + Illiteracy      1     0.131 25.058 -22.541
## - HS.Grad         1     2.949 28.138 -20.745
## - Frost           1     4.479 29.668 -18.097
```

```
## - Murder           1    32.877 58.067  15.478
##
## Step:  AIC=-27.02
## Life.Exp ~ Income + Murder + Frost + HS.Grad + Log.Population
##
##                     Df Sum of Sq    RSS      AIC
## - Income            1     0.011 22.921 -28.998
## <none>                          22.911 -27.021
## + Income.2          1     0.579 22.331 -26.302
## + Log.Area          1     0.207 22.704 -25.475
## + Illiteracy        1     0.052 22.859 -25.134
## + HS.Grad.2         1     0.009 22.901 -25.042
## - Frost             1     2.107 25.017 -24.623
## - Log.Population    1     2.279 25.189 -24.280
## - HS.Grad           1     4.436 27.347 -20.172
## - Murder            1    33.706 56.616  16.214
##
## Step:  AIC=-29
## Life.Exp ~ Murder + Frost + HS.Grad + Log.Population
##
##                     Df Sum of Sq    RSS      AIC
## <none>                          22.921 -28.998
## + Log.Area          1     0.216 22.705 -27.471
## + Illiteracy        1     0.052 22.870 -27.111
## + Income.2          1     0.034 22.887 -27.073
## + HS.Grad.2         1     0.012 22.909 -27.024
## + Income            1     0.011 22.911 -27.021
## - Frost             1     2.214 25.135 -26.387
## - Log.Population    1     2.450 25.372 -25.920
## - HS.Grad           1     6.959 29.881 -17.741
## - Murder            1    34.109 57.031  14.578


##
## Call:
## lm(formula = Life.Exp ~ Murder + Frost + HS.Grad + Log.Population,
##     data = state.data)
##
## Coefficients:
##    (Intercept)         Murder          Frost        HS.Grad  Log.Population
##      68.720810      -0.290016      -0.005174       0.054550        0.246836
```

Notice that our model selected by `step()` is not the same model we obtained when we started with the biggest model and removed things based on p-values.

The log-likelihood is only defined up to an additive constant, and there are different conventional constants used. This is more annoying than anything

because all we care about for model selection is the difference between AIC values of two models and the additive constant cancels. The only time it matters is when you have two different ways of extracting the AIC values. Recall the model we fit using the top-down approach was

```r
# m1 was
m1 <- lm(Life.Exp ~ Income + Murder + Frost + Income.2, data = state.data)
AIC(m1)
```

```
## [1] 121.4293
```

and the model selected by the stepwise algorithm was

```r
m3 <- lm(Life.Exp ~ Murder + Frost + HS.Grad + Log.Population, data = state.data)
AIC(m3)
```

```
## [1] 114.8959
```

Because `step()` and `AIC()` are following different conventions the absolute value of the AICs are different, but the difference between the two is constant no matter which function we use.

First we calculate the difference using the AIC() function:

```r
AIC(m1) - AIC(m3)
```

```
## [1] 6.533434
```

and next we use `add1()` on both models to see what the AIC values for each.

```r
add1(m1, scope=biggest)
```

```
## Single term additions
##
## Model:
## Life.Exp ~ Income + Murder + Frost + Income.2
##                 Df Sum of Sq    RSS     AIC
## <none>                       26.121 -22.465
## Log.Population   1   0.10296 26.018 -20.662
## Illiteracy       1   0.10097 26.020 -20.658
## HS.Grad          1   2.79527 23.326 -26.124
## Log.Area         1   2.36019 23.761 -25.200
## HS.Grad.2        1   2.79698 23.324 -26.127
```

```
add1(m3, scope=biggest)
```

```
## Single term additions
##
## Model:
## Life.Exp ~ Murder + Frost + HS.Grad + Log.Population
##             Df Sum of Sq    RSS     AIC
## <none>                   22.921 -28.998
## Income       1  0.010673 22.911 -27.021
## Illiteracy   1  0.051595 22.870 -27.111
## Log.Area     1  0.215741 22.706 -27.471
## HS.Grad.2    1  0.011894 22.909 -27.024
## Income.2     1  0.034356 22.887 -27.073
```

Using these results, we can calculate the difference in AIC values to be the same as we calculated before

$$-22.465 - -28.998 = -22.465 + 28.998$$
$$= 6.533$$

9.4 Exercises

1. Consider the **prostate** data from the **faraway** package. The variable **lpsa** is a measurement of a prostate specific antigen which higher levels are indicative of prostate cancer. Use **lpsa** as the response and all the other variables as predictors (no interactions). Determine the "best" model using:

   a. Backward elimination using the analysis of variance F-statistic as the criteria.
   b. Forward selection using AIC as the criteria.

2. Again from the **faraway** package, use the **divusa** which has divorce rates for each year from 1920-1996 along with other population information for each year. Use **divorce** as the response variable and all other variables as the predictors.

   a. Determine the best model using stepwise selection starting from the intercept only model and the most complex model being all main effects (no interactions). Use the F-statistic to determine significance. Note: add1(), drop1(), and step() allow an option of test='F' to use an F-test instead of AIC.
   b. Following the stepwise selection, comment on the relationship between p-values used and the AIC difference observed. Do the AIC rules of thumb match the p-value interpretation?

Chapter 10

Mixed Effects Models

```r
library(tidyverse)   # dplyr, tidyr, ggplot
library(stringr)     # string manipulation stuff
library(latex2exp)   # for LaTeX mathematical notation

library(lme4)         # Our primary analysis routine
library(lmerTest)     # A user friendly interface to lme4 that produces p-values
library(emmeans)      # For all of my pairwise contrasts
library(car)          # For bootstrap Confidence/Prediction Intervals

# library(devtools)
# install_github('dereksonderegger/dsData')  # datasets I've made; only install once...
library(dsData)
```

The assumption of independent observations is often not supported and dependent data arises in a wide variety of situations. The dependency structure could be very simple such as rabbits within a litter being correlated and the litters being independent. More complex hierarchies of correlation are possible. For example we might expect voters in a particular part of town (called a precinct) to vote similarly, and particular districts in a state tend to vote similarly as well, which might result in a precinct / district / state hierarchy of correlation.

10.1 Block Designs

Often there are covariates in the experimental units that are known to affect the response variable and must be taken into account. Ideally an experimenter can group the experimental units into blocks where the within block variance

is small, but the block to block variability is large. For example, in testing a drug to prevent heart disease, we know that gender, age, and exercise levels play a large role. We should partition our study participants into gender, age, and exercise groups and then randomly assign the treatment (placebo vs drug) within the group. This will ensure that we do not have a gender, age, and exercise group that has all placebo observations.

Often blocking variables are not the variables that we are primarily interested in, but must nevertheless be considered. We call these nuisance variables. We already know how to deal with these variables by adding them to the model, but there are experimental designs where we must be careful because the experimental treatments are *nested*.

Example 1. An agricultural field study has three fields in which the researchers will evaluate the quality of three different varieties of barley. Due to how they harvest the barley, we can only create a maximum of three plots in each field. In this example we will block on field since there might be differences in soil type, drainage, etc from field to field. In each field, we will plant all three varieties so that we can tell the difference between varieties without the block effect of field confounding our inference. In this example, the varieties are nested within the fields.

	Field 1	Field 2	Field 3
Plot 1	Variety A	Variety C	Variety B
Plot 2	Variety B	Variety A	Variety C
Plot 3	Variety C	Variety B	Variety A

Example 2. We are interested in how a mouse responds to five different materials inserted into subcutaneous tissue to evaluate the materials' use in medicine. Each mouse can have a maximum of 3 insertions. Here we will block on the individual mice because even lab mice have individual variation. We actually are not interested in estimating the effect of the mice because they aren't really of interest, but the mouse block effect should be accounted for before we make any inferences about the materials. Notice that if we only have one insertion per mouse, then the mouse effect will be confounded with materials.
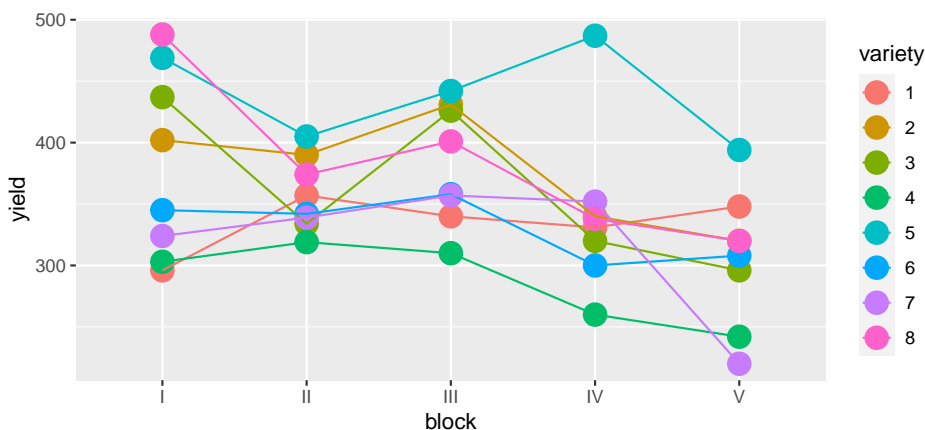
10.2   Randomized   Complete   Block   Design (RCBD)

The dataset `oatvar` in the faraway library contains information about an experiment on eight different varieties of oats. The area in which the experiment was done had some systematic variability and the researchers divided the area up into five different blocks in which they felt the area inside a block was uniform

while acknowledging that some blocks are likely superior to others for growing crops. Within each block, the researchers created eight plots and randomly assigned a variety to a plot. This type of design is called a Randomized Complete Block Design (RCBD) because each block contains all possible levels of the factor of primary interest.

```
data('oatvar', package='faraway')
ggplot(oatvar, aes(y=yield, x=block, color=variety)) +
    geom_point(size=5) +
    geom_line(aes(x=as.integer(block))) # connect the dots
```


While there is one unusual observation in block IV, there doesn't appear to be a blatant interaction. We will consider the interaction shortly. For the main effects model of yield ~ block + variety we have $p = 12$ parameters and 28 residual degrees of freedom because

$$
\begin{aligned}
df_\epsilon &= n - p \\
&= n - (1 + [(I - 1) + (J - 1)]) \\
&= 40 - (1 + [(5 - 1) + (8 - 1)]) \\
&= 40 - 12 \\
&= 28
\end{aligned}
$$

```
m1 <- lm( yield ~ block + variety, data=oatvar)
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## block      4  33396  8348.9  6.2449  0.001008 **
```

```
## variety    7  77524 11074.8   8.2839 1.804e-05 ***
## Residuals 28  37433   1336.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# plot(m1)       # check diagnostic plots - they are fine...
```

Because this is an orthogonal design, the sums of squares doesn't change regardless of which order we add the factors, but if we remove one or two observations, they would.
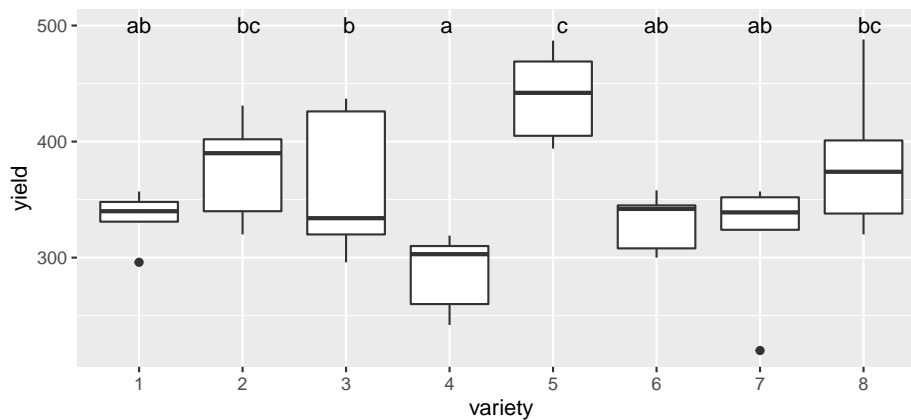
In determining the significance of `variety` the above F-value and p-value is correct. We have 40 observations (5 per variety), and after accounting for the model structure (including the extraneous blocking variable), we have 28 residual degrees of freedom.

But the F-value and p-value for testing if `block` is significant is nonsense! Imagine that variety didn't matter we just have 8 replicate samples per block, but these aren't true replicates, they are what is called *pseudoreplicates*. Imagine taking a sample of $n = 3$ people and observing their height at 1000 different points in time during the day. You don't have 3000 data points for estimating the mean height in the population, you have 3. Unless we account for the this, the inference for the block variable is wrong. In this case, we only have one observation for each block, so we can't do any statistical inference at the block scale!

Fortunately in this case, we don't care about the blocking variable and including it in the model was simply guarding us in case there was a difference, but I wasn't interested in estimating it. If the only covariate we care about is the most deeply nested effect, then we can do the usual analysis and recognize the p-value for the blocking variable is nonsense, and we don't care about it.

```
# Ignore any p-values regarding block, but I'm happy with the analysis for variety
letter_df <- emmeans(m1, ~variety) %>%
  multcomp::cld(Letters=letters) %>%
  dplyr::select(variety, .group) %>%
  mutate(yield = 500)

ggplot(oatvar, aes(x=variety, y=yield)) +
  geom_boxplot() +
  geom_text( data=letter_df, aes(label=.group) )
```

However it would be pretty sloppy to not do the analysis correctly because our blocking variable might be something we care about.

Once correct way to model these data are using hierarchical models which are created by having multiple error terms that can be introduced. In many respects, the random effects structure provides an extremely flexible framework to consider many of the traditional experimental designs as well as many non-traditional designs with the benefit of more easily assessing variability at each hierarchical level.

Mixed effects models combine what we call "fixed" and "random" effects.

Fixed effects	Unknown constants that we wish to estimate from the model and could be similarly estimated in subsequent experimentation. The research is interested in these particular levels.
Random effects	Random variables sampled from a population which cannot be observed in subsequent experimentation. The research is not interested in these particular levels, but rather how the levels vary from sample to sample.

For example, in a rabbit study that examined the effect of diet on the growth of domestic rabbits and we had 10 litters of rabbits and used the 3 most similar from each litter to test 6 different diets. Here, the 6 different diets are fixed effects because they are not randomly selected from a population, these exact same diets can be further studied, and these are the diets we are interested it. The litters of rabbits and the individual rabbits are randomly selected from populations, cannot be exactly replicated in future studies, and we are not interested in the individual litters but rather what the variability is between individuals and between litters.

Often random effects are not of primary interest to the researcher, but must

be considered. Blocking variables are random effects because the arise from a random sample of possible blocks that are potentially available to the researcher.

Mixed effects models are models that have both fixed and random effects. We will first concentrate on understanding how to address a model with two sources error and then complicate the matter with fixed effects.

10.3   Review of Maximum Likelihood Methods

Recall that the likelihood function is the function links the model parameters to the data and is found by taking the probability density function and interpreting it as a function of the parameters instead of the a function of the data. Loosely, the probability function tells us what outcomes are most probable, with the height of the function telling us which values (or regions of values) are most probable given a set of parameter values. The higher the probability function, the higher the probability of seeing that value (or data in that region). The likelihood function turns that relationship around and tells us what parameter values are most likely to have generated the data we have, again with the parameter values with a higher likelihood value being more "likely".

The likelihood function for a sample $y_i \overset{iid}{\sim} N\left(X_{i,\cdot}\beta, \sigma\right)$ can be written as a function of our parameters β and σ^2 then we have defined our likelihood function

$$L\left(\beta, \sigma^2 | y_1, \ldots, y_n\right) = \frac{1}{(2\pi)^{n/2}\left[\det\left(\Omega\right)\right]^{1/2}} \exp\left[-\frac{1}{2}\left(y - X\beta\right)^T \Omega^{-1}\left(y - X\beta\right)\right]$$

where the variance/covariance matrix is $\Omega = \sigma I_n$.

We can use to this equation to find the maximum likelihood estimators by either taking the derivatives and setting them equal to zero and solving for the parameters or by using numerical methods. In the normal case, we can find the maximum likelihood estimators (MLEs) using the derivative trick and we find that

$$\widehat{\beta}_{MLE} = \left(X^T \Omega^{-1} X\right)^{-1} X^T \Omega^{-1} y$$

and

$$\widehat{\sigma}^2_{MLE} = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \widehat{y}\right)^2$$

and where $\widehat{y}_i = X_{i,\cdot}\widehat{\beta}$ we notice that this is not our usual estimator $\widehat{\sigma}^2 = s^2$ where s^2 is the sample variance. It turns out that the MLE estimate of σ^2 is biased (the correction is to divide by $n - 1$ instead of n). This is normally not an issue if our sample size is large, but with a small sample, the bias is not insignificant.

Notice if we happened to know that $\mu = 0$, then we could use

$$\hat{\sigma}^2_{MLE} = \frac{1}{n} \sum_{i=1}^{n} y_i^2$$

and this would be unbiased for σ^2.

In general (a not just in the normal case above) the *Likelihood Ratio Test* (LRT) provides a way for us to compare two nested models. Given m_0 which is a simplification of m_1 then we could calculate the likelihoods functions of the two models $L(\theta_0)$ and $L(\theta_1)$ where θ_0 is a vector of parameters for the null model and θ_1 is a vector of parameter for the alternative. Let $\hat{\theta}_0$ be the maximum likelihood estimators for the null model and $\hat{\theta}_1$ be the maximum likelihood estimators for the alternative. Finally we consider the value of

$$D = -2 * \log \left[\frac{L\left(\hat{\theta}_0\right)}{L\left(\hat{\theta}_1\right)} \right]$$
$$= -2 \left[\log L\left(\hat{\theta}_0\right) - \log L\left(\hat{\theta}_1\right) \right]$$

Under the null hypothesis that m_0 is the true model, the $D \overset{\cdot}{\sim} \chi^2_{p_1 - p_0}$ where $p_1 - p_0$ is the difference in number of parameters in the null and alternative models. That is to say that asymptotically D has a Chi-squared distribution with degrees of freedom equal to the difference in degrees of freedom of the two models.

We could think of $L\left(\hat{\theta}_0\right)$ as the maximization of the likelihood when some parameters are held constant (at zero) and all the other parameters are vary. But we are not required to hold it constant at zero. We could chose any value of interest and perform a LRT.

Because we often regard a confidence interval as the set of values that would not be rejected by a hypothesis test, we could consider a sequence of possible values for a parameter and figure out which would not be rejected by the LRT. In this fashion we can construct confidence intervals for parameter values.

Unfortunately all of this hinges on the asymptotic distribution of D and often this turns out to be a poor approximation. In simple cases more exact tests can be derived (for example the F-tests we have used prior) but sometimes nothing better is currently known. Another alternative is to use resampling methods for the creation of confidence intervals or p-values.

10.4 1-way ANOVA with a random effect

We first consider the simplest model with two sources of variability, a 1-way ANOVA with a random factor covariate

$$y_{ij} = \mu + \gamma_i + \epsilon_{ij}$$

where $\gamma_i \overset{iid}{\sim} N\left(0, \sigma_\gamma^2\right)$ and $\epsilon_{ij} \overset{iid}{\sim} N\left(0, \sigma_\epsilon^2\right)$. This model could occur, for example, when looking at the adult weight of domestic rabbits where the random effect is the effect of litter and we are interested in understanding how much variability there is between litters $\left(\sigma_\gamma^2\right)$ and how much variability there is within a litter $\left(\sigma_\epsilon^2\right)$. Another example is the the creation of computer chips. Here a single wafer of silicon is used to create several chips and we might have wafer-to-wafer variability and then within a wafer, you have chip-to-chip variability.

First we should think about what the variances and covariances are for any two observations.
$$\begin{aligned} Var\left(y_{ij}\right) &= Var\left(\mu + \gamma_i + \epsilon_{ij}\right) \\ &= Var\left(\mu\right) + Var\left(\gamma_i\right) + Var\left(\epsilon_{ij}\right) \\ &= 0 + \sigma_\gamma^2 + \sigma_\epsilon^2 \end{aligned}$$

and $Cov\left(y_{ij}, y_{ik}\right) = \sigma_\gamma^2$ because the two observations share the same litter γ_i. For two observations in different litters, the covariance is 0. These relationships induce a correlation on observations within the same litter of

$$\rho = \frac{\sigma_\gamma^2}{\sigma_\gamma^2 + \sigma_\epsilon^2}$$

For example, suppose that we have $I = 3$ litters and in each litter we have $J = 3$ rabbits per litter. Then the variance-covariance matrix looks like

$$\Omega = \begin{bmatrix} \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 & \sigma_\gamma^2 & . & . & . & . & . & . \\ \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 & . & . & . & . & . & . \\ \sigma_\gamma^2 & \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 & . & . & . & . & . & . \\ . & . & . & \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 & \sigma_\gamma^2 & . & . & . \\ . & . & . & \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 & . & . & . \\ . & . & . & \sigma_\gamma^2 & \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 & . & . & . \\ . & . & . & . & . & . & \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 & \sigma_\gamma^2 \\ . & . & . & . & . & . & \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 & \sigma_\gamma^2 \\ . & . & . & . & . & . & \sigma_\gamma^2 & \sigma_\gamma^2 & \sigma_\gamma^2 + \sigma_\epsilon^2 \end{bmatrix}$$

Substituting this new variance-covariance matrix into our likelihood function, we now have a likelihood function which we can perform our usual MLE tricks with.

In the more complicated situation where we have a full mixed effects model, we could write
$$y = X\beta + Z\gamma + \epsilon$$

where X is the design matrix for the fixed effects, β is the vector of fixed effect coefficients, Z is the design matrix for random effects, γ is the vector of random effects such that $\gamma_i \overset{iid}{\sim} N\left(0, \sigma_\gamma^2\right)$ and finally ϵ is the vector of error terms such that $\epsilon_{ij} \overset{iid}{\sim} N\left(0, \sigma_\epsilon^2\right)$. Notice in our rabbit case

$$Z = \begin{bmatrix} 1 & \cdot & \cdot \\ 1 & \cdot & \cdot \\ 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \\ \cdot & 1 & \cdot \\ \cdot & 1 & \cdot \\ \cdot & \cdot & 1 \\ \cdot & \cdot & 1 \\ \cdot & \cdot & 1 \end{bmatrix} \quad ZZ^T = \begin{bmatrix} 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 \end{bmatrix}$$

which makes it easy to notice

$$\Omega = \sigma_\gamma^2 ZZ^T + \sigma_\epsilon^2 I$$

In practice we tend to have relatively small numbers of block parameters and thus have a small number of observations in which to estimate σ_γ^2 which means that the biased nature of MLE estimates will be sub-optimal. If we knew that $X\beta = 0$ we could use that fact and have an unbiased estimate of our variance parameters. Because X is known, we can find linear functions k such that $k^T X = 0$. We can form a matrix K that represents all of these possible transformations and we notice that

$$K^T y \sim N\left(K^T X\beta, \, K^T \Omega K\right) = N\left(0, K^T \Omega K\right)$$

and perform our maximization on this transformed set of data. Once we have our unbiased estimates of σ_γ^2 and σ_ϵ^2, we can substitute these back into the untransformed likelihood function and find the MLEs for β. This process is called Restricted Maximum Likelihood (REML) and is generally preferred over the variance component estimates found simply maximizing the regular likelihood function. As usual, if our experiment is balanced these complications aren't necessary as the REML estimates of β are usually the same as the ML estimates.

Our first example comes from an experiment to test the paper brightness as affected by the shift operator. The data has 20 observations with 4 different operators. Each operator had 5 different observations made. The data set is `pulp` in the package `faraway`. We will first analyze this using a fixed-effects one-way ANOVA, but we will use a different model representation. Instead of using the first operator as the reference level, we will use the sum-to-zero constraint (to make it easier to compare with the output of the random effects model).

```
data('pulp', package='faraway')
ggplot(pulp, aes(x=operator, y=bright)) + geom_point()
```
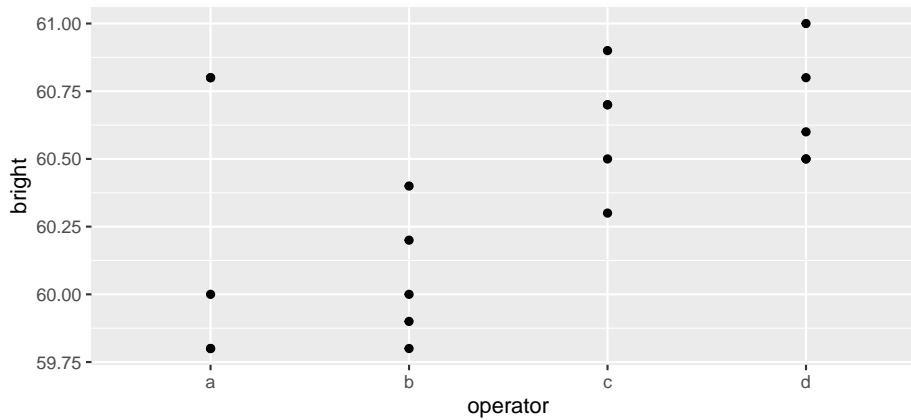
```r
# set the contrasts to sum-to-zero constraint
options(contrasts=c('contr.sum', 'contr.poly'))
m <- lm(bright ~ operator, data=pulp)
summary(m)
```

```
##
## Call:
## lm(formula = bright ~ operator, data = pulp)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.440 -0.195 -0.070  0.175  0.560
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 60.40000    0.07289 828.681   <2e-16 ***
## operator1   -0.16000    0.12624  -1.267    0.223
## operator2   -0.34000    0.12624  -2.693    0.016 *
## operator3    0.22000    0.12624   1.743    0.101
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.326 on 16 degrees of freedom
## Multiple R-squared:  0.4408, Adjusted R-squared:  0.3359
## F-statistic: 4.204 on 3 and 16 DF,  p-value: 0.02261
```

```r
coef(m)
```

```
## (Intercept)   operator1   operator2   operator3
##       60.40       -0.16       -0.34        0.22
```

The sum-to-zero constraint forces the operator parameters to sum to zero so we can find the value of the fourth operator as $operator4 = -(-0.16 - 0.34 + 0.22) = 0.28$

To fit the random effects model we will use the package `lmerTest` which is a nicer user interface to the package `lme4`. The reason we won't use `lme4` directly is that the authors of `lme4` refuse to calculate p-values. The reason for this is that in mixed models it is not always clear what the appropriate degrees of freedom are for the residuals, and therefore we don't know what the appropriate t-distribution is to compare the t-values to. In simple balanced designs the degrees of freedom can be calculated, but in complicated unbalanced designs the appropriate degrees of freedom is not known and all proposed heuristic methods (including what is calculated by SAS) can fail spectacularly in certain cases. The authors of `lme4` are adamant that until robust methods are developed, they prefer to not calculate any p-values. There are other packages out there that recognize that we need approximate p-values and the package `lmerTest` provides reasonable answers that match was SAS calculates.

```
m2 <- lmer( bright ~ 1 + (1|operator), data=pulp )
summary(m2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: bright ~ 1 + (1 | operator)
##    Data: pulp
##
## REML criterion at convergence: 18.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.4666 -0.7595 -0.1244  0.6281  1.6012
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  operator (Intercept) 0.06808  0.2609
##  Residual             0.10625  0.3260
## Number of obs: 20, groups:  operator, 4
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)  60.4000     0.1494  3.0000   404.2 3.34e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice that the estimate of the fixed effect (the overall mean) is the same in the fixed-effects ANOVA and in the mixed model. However the fixed effects

ANOVA estimates the effect of each operator while the mixed model is interested in estimating the variance between operators. In the model statement the (1|operator) denotes the random effect and this notation tells us to fit a model with a random intercept term for each operator. Here the variance associated with the operators is $\sigma_\gamma^2 = 0.068$ while the "pure error" is $\sigma_\epsilon^2 = 0.106$. The column for standard deviation is not the variability associated with our estimate, but is simply the square-root of the variance terms σ_γ and σ_ϵ. This was fit using the REML method.

We might be interested in the estimated effect of each operator

```
ranef(m2)
```

```
## $operator
##    (Intercept)
## a  -0.1219403
## b  -0.2591231
## c   0.1676679
## d   0.2133955
##
## with conditional variances for "operator"
```

These effects are smaller than the values we estimated in the fixed effects model due to distributional assumption that penalizes large deviations from the mean. In general, the estimated random effects are of smaller magnitude than the effect size estimated using a fixed effect model.

```
# reset the contrasts to the default
options(contrasts=c("contr.treatment", "contr.poly" ))
```

10.5   Blocks as Random Variables

Blocks are properties of experimental designs and usually we are not interested in the block levels *per se* but need to account for the variability introduced by them.

Recall the agriculture experiment in the dataset `oatvar` from the `faraway` package. We had 8 different varieties of oats and we had 5 different fields (which we called blocks). Because of limitations on how we plant, we could only divide the blocks into 8 plots and in each plot we planted one of the varieties.

```
data('oatvar', package='faraway')
ggplot(oatvar, aes(y=yield, x= variety)) +
  geom_point() +
  facet_wrap(~block, labeller=label_both)
```
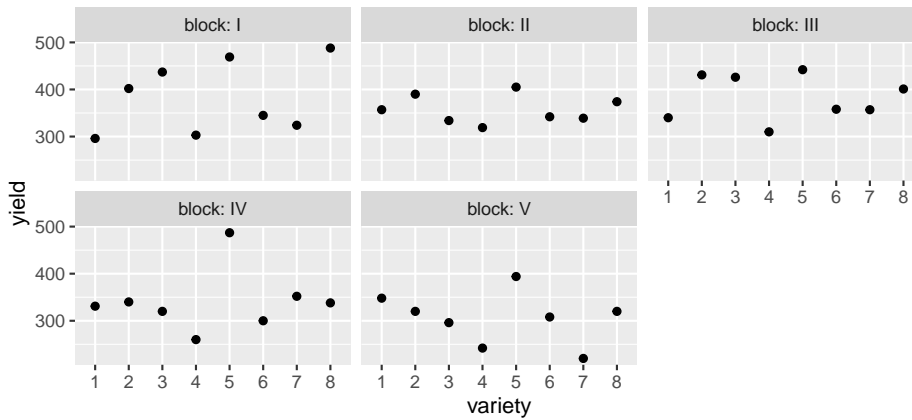
In this case, we don't really care about these particular fields (blocks) and would prefer to think about these as a random sample of fields that we might have used in our experiment. The analysis to compare these simple model (without variety) to the complex model it should use ML because the fixed effects are different between the two models and thus the K matrix used in REML fits are different.

```
model.0 <- lmer( yield ~ (1|block), data=oatvar)
model.1 <- lmer( yield ~ variety + (1|block), data=oatvar)

# By default anova() will always refit the models using ML assuming you want to
# compare models with different fixed effects. Use refit=FALSE to suppress this.
anova(model.0, model.1)
```

```
## refitting model(s) with ML (instead of REML)

## Data: oatvar
## Models:
## model.0: yield ~ (1 | block)
## model.1: yield ~ variety + (1 | block)
##          npar    AIC    BIC  logLik deviance Chisq Df Pr(>Chisq)
## model.0     3 446.94 452.01 -220.47   440.94
## model.1    10 421.67 438.56 -200.84   401.67 39.27  7  1.736e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice that this is doing a Likelihood Ratio Test

$$-2 * \log\left(\frac{L_0}{L_a}\right) = -2\left(\log(L_0) - \log(L_a)\right) = -2*(-220.47 - -200.56) = 39.24$$

Recall that we don't really like to trust Likelihood Ratio Tests because they depend on the asymptotic distribution of the statistic (as sample size increases)

and that convergence is pretty slow. Another, and usually better option, is to perform an F-test with the numerator degrees of freedom estimated using either the *Satterthwaite* or *Kenward-Roger* methods. To do this, we'll use the `anova()` command with just a single model.

```r
# Do an F test for the fixed effects using the similar degree of freedom
# approximations done by SAS
#
# anova(model.1, ddf='lme4')  # don't return the p-value because we don't trust it!
# anova(model.1, ddf='Satterthwaite')  # Use Satterthwaite
# anova(model.1, ddf='Kenward-Roger')  # Use Kenward-Roger
anova(model.1)   # default is Satterthwaite's degrees of freedom
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##          Sum Sq Mean Sq NumDF DenDF F value     Pr(>F)
## variety  77524   11075     7    28  8.2839 1.804e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Unsurprisingly given our initial thoughts about the data, it looks like `variety` is a statistically significant covariate.

There is quite a bit of debate among statisticians about how to calculate the denominator degrees of freedom and which method is preferred in different scenarios. By default, `lmerTest` uses Satterthwaite's method, but 'Kenward-Roger' is also allowed. In this case, the two methods produces the same estimated denominator degrees of freedom and the same p-value.

To consider if the random effect should be included in the model, we will turn to the Likelihood Ratio test. The following examines all single term deletions to the random effects structure. In our case, this is just considering removing `(1|block)`

```r
# Each line tests if a random effect can be removed or reduced to a simpler random eff
# Something like (1|Group) will be tested if it can be removed.
# Something like (1+Trt | Group) will be tested if it can be reduced to (1 | Group)
ranova(model.1)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## yield ~ variety + (1 | block)
##             npar  logLik    AIC    LRT Df Pr(>Chisq)
## <none>        10 -170.68 361.35
## (1 | block)    9 -175.08 368.16 8.8065  1   0.003002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here, both the AIC and LRT suggest that the random effect of block is appropriate to include in the model.

Now that we have chosen our model, we can examine is model.

```
summary(model.1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: yield ~ variety + (1 | block)
##    Data: oatvar
##
## REML criterion at convergence: 341.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.7135 -0.5503 -0.1280  0.4862  2.1756
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  block    (Intercept)  876.5   29.61
##  Residual             1336.9   36.56
## Number of obs: 40, groups:  block, 5
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)   334.40      21.04   15.26  15.894 6.64e-11 ***
## variety2       42.20      23.12   28.00   1.825   0.0787 .
## variety3       28.20      23.12   28.00   1.219   0.2328
## variety4      -47.60      23.12   28.00  -2.058   0.0490 *
## variety5      105.00      23.12   28.00   4.541 9.73e-05 ***
## variety6       -3.80      23.12   28.00  -0.164   0.8707
## variety7      -16.00      23.12   28.00  -0.692   0.4947
## variety8       49.80      23.12   28.00   2.154   0.0400 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) varty2 varty3 varty4 varty5 varty6 varty7
## variety2 -0.550
## variety3 -0.550  0.500
## variety4 -0.550  0.500  0.500
## variety5 -0.550  0.500  0.500  0.500
## variety6 -0.550  0.500  0.500  0.500  0.500
## variety7 -0.550  0.500  0.500  0.500  0.500  0.500
## variety8 -0.550  0.500  0.500  0.500  0.500  0.500  0.500
```

We start with the Random effects. This section shows us the *block-to-block* variability (and the square root of that, the Standard Deviation) as well as the "pure-error", labeled `residuals`, which is an estimate of the variability associated with two different observations (after the difference in variety is accounted for) planted *within* the same block. For this we see that block-to-block variability is only slightly smaller than the within block variability.

Why do we care about this? This actually tells us quite a lot about the spatial variability. Because yield is affected by soil nutrients, micro-climate, soil water availability, etc, I expect that two identical seedlings planted in slightly different conditions will have slightly different yields. By examining how the yield changes over small distances (the residual within block variability) vs how it changes over long distances (block to block variability) we can get a sense as to the scale at which these background lurking processes operate.

Next we turn to the fixed effects. These will be the offsets from the reference group, as we've typically worked with. Here we see that varieties 2,5, and 8 are the best performers (relative to variety 1),

We are certain that there are differences among the varieties, and we should look at all of the pairwise contrasts among the variety levels. As usual we could use the package `emmeans`, which automates much of this (and uses `lmerTest` produced p-values for the tests).

```
LetterResults <- emmeans::emmeans( model.1, ~ variety) %>%
  multcomp::cld(Letters=letters)
LetterResults
```

```
##   variety emmean SE   df lower.CL upper.CL .group
##   4          287 21 15.2      242      332  a
##   7          318 21 15.2      274      363  ab
##   6          331 21 15.2      286      375  ab
##   1          334 21 15.2      290      379  ab
##   3          363 21 15.2      318      407   b
##   2          377 21 15.2      332      421   bc
##   8          384 21 15.2      339      429   bc
##   5          439 21 15.2      395      484    c
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 8 estimates
## significance level used: alpha = 0.05
```

As usual we'll join this information into the original data table and then make a nice summary graph.

```
LetterResults <- LetterResults %>%
  mutate(LetterHeight=500,  .group = str_trim(.group))

oatvar %>%
  mutate(variety = fct_reorder(variety, yield)) %>%
  ggplot( aes(x=variety, y=yield)) +
  geom_point(aes(color=block)) +
  geom_text(data=LetterResults, aes(label=.group, y=LetterHeight))
```
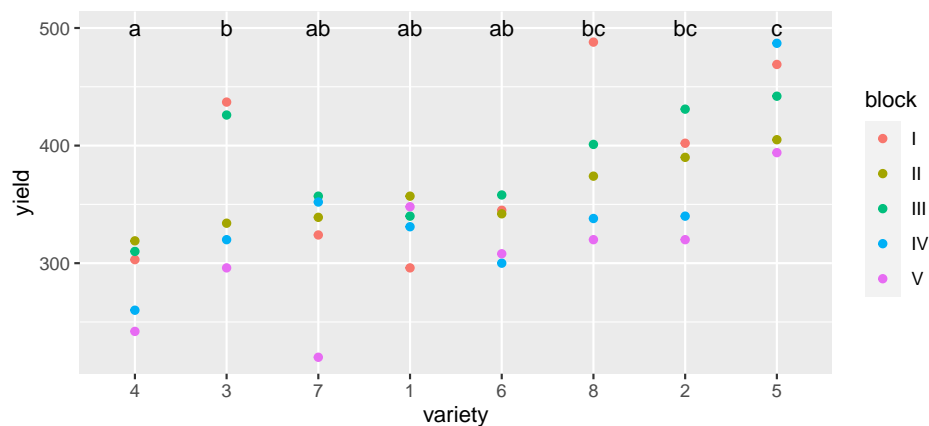

We'll consider a second example using data from the pharmaceutical industry. We are interested in 4 different processes (our treatment variable) used in the biosynthesis and purification of the drug penicillin. The biosynthesis requires a nutrient source (corn steep liquor) as a nutrient source for the fungus and the nutrient source is quite variable. Each batch of the nutrient is is referred to as a 'blend' and each blend is sufficient to create 4 runs of penicillin. We avoid confounding our biosynthesis methods with the blend by using a Randomized Complete Block Design and observing the yield of penicillin from each of the four methods (A,B,D, and D) in each blend.

```
data(penicillin, package='faraway')

ggplot(penicillin, aes(y=yield, x=treat)) +
  geom_point() +
  facet_wrap( ~ blend, ncol=5)
```
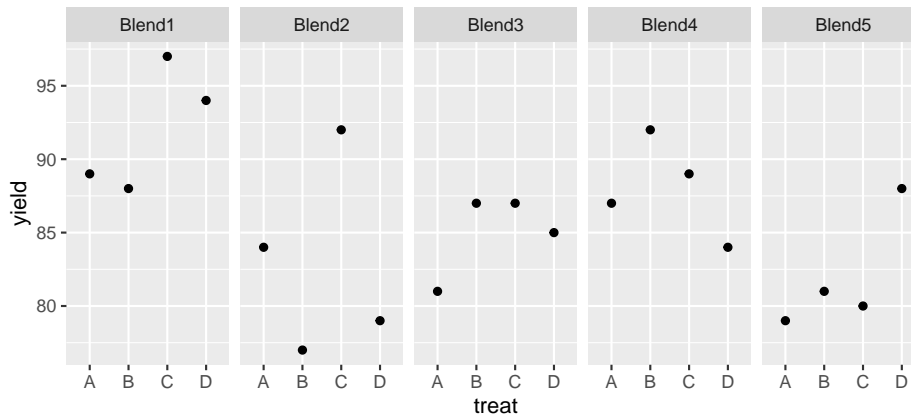
It looks like there is definitely a `Blend` effect (e.g. Blend1 is much better than Blend5) but it isn't clear that there is a treatment effect.

```
model.0 <- lmer(yield ~  1    + (1 | blend), data=penicillin)
model.1 <- lmer(yield ~ treat + (1 | blend), data=penicillin)
# anova(model.0, model.1)  # Analysis using a LRT. Not my preference for analysis
anova(model.1)  # Tests using an F-test with an approximated degrees of freedom
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## treat      70  23.333     3    12  1.2389 0.3387
```

It looks like we don't have a significant effect of the treatments. Next we'll examine the simple model to understand the variability.

```
# Test if we should remove the (1|blend) term using either AIC or Likelihood Ratio Tes
ranova(model.1)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## yield ~ treat + (1 | blend)
##              npar  logLik    AIC    LRT Df Pr(>Chisq)
## <none>          6 -51.915 115.83
## (1 | blend)     5 -53.296 116.59 2.7629  1    0.09647 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Blend looks marginally significant, but it might still be interesting to compare the variability between blends to variability within blends.

```
# summary(model.0)                  # Shows everything...
summary(model.0)[['varcor']]       # Just the sigma terms for each random term
```

```
##  Groups    Name        Std.Dev.
##  blend     (Intercept) 3.4010
##  Residual              4.4422
```

We see that the noise is more in the within blend (labeled residual here) rather than the between blends. If my job were to understand the variability and figure out how to improve production, this suggests that variability is introduced at the blend level *and* at the run level, but the run level is more important.

10.6 Nested Effects

When the levels of one factor vary only within the levels of another factor, that factor is said to be nested. For example, when measuring the performance of workers at several job locations, if the workers only work at one site, then the workers are nested within site. If the workers work at more than one location, we would say that workers are *crossed* with site.

We've already seen a number of nested designs when we looked at split plot designs. Recall the `AgData` set that I made up that simulated an agricultural experiment with 8 plots and 4 subplots per plot. We applied an irrigation treatment at the plot level and a fertilizer treatment at the subplot level. I actually have 5 replicate observations per subplot.
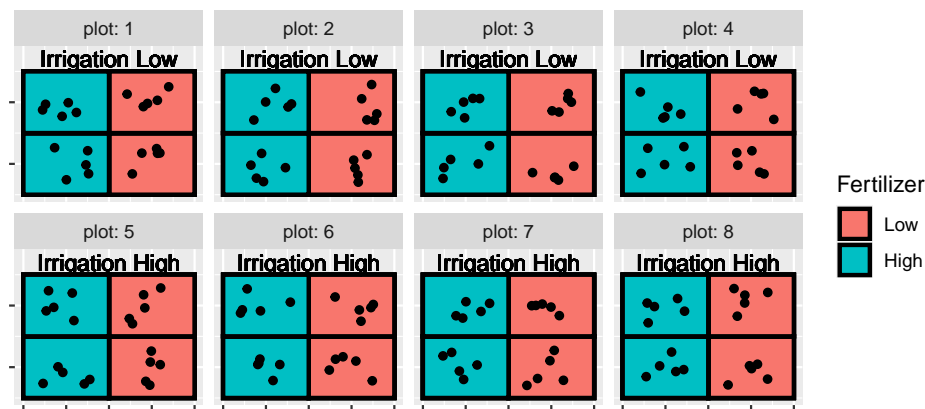
So all together we have 8 plots, 32 subplots, and 5 replicates per subplot. When I analyze the fertilizer, I have 32 experimental units (the thing I have applied my treatment to), but when analyzing the effect of irrigation, I only have 8 experimental units. In other words, I should have 8 random effects for plot, and 32 random effects for subplot.

```
# The following model definitions are equivalent
model <- lmer(yield ~ Irrigation + Fertilizer + (1|plot) + (1|plot:subplot), data=AgData)
model <- lmer(yield ~ Irrigation + Fertilizer + (1|plot/subplot), data=AgData)
anova(model)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##             Sum Sq Mean Sq NumDF  DenDF F value    Pr(>F)
## Irrigation  3.4769  3.4769     1  5.9999  3.4281    0.1136
## Fertilizer 31.3824 31.3824     1 22.9999 30.9420 1.169e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As we saw before, the effect of irrigation is not significant and the fertilizer effect is highly significant. We'll remove the irrigation covariate and refit the model.

We designed our experiment assuming that both `plot` and `subplot` were possibly important. Many statisticians would argue that because that was how we designed the experiment, we should necessarily keep that structure when analyzing the data. This is particularly compelling considering that the Irrigation and Fertilizer were applied on different scales. However it is still fun to look at the statistical significance of the random effects.

```
model <- lmer(yield ~ Fertilizer + (1|plot/subplot), data=AgData)
ranova(model)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## yield ~ Fertilizer + (1 | subplot:plot) + (1 | plot)
##                    npar  logLik    AIC     LRT Df Pr(>Chisq)
## <none>                5 -286.32 582.64
## (1 | subplot:plot)    4 -369.99 747.97 167.334  1  < 2.2e-16 ***
## (1 | plot)            4 -293.00 594.01  13.367  1  0.0002561 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Each row (except the `none`) is testing the exclusion of a random effect. The first being the 32 subplots, and the second being the 8 plots. Both are highly statistically significant. To assess the practical significance of subplots and plots, we need to look at the variance terms:

```
summary(model)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
```

```
## lmerModLmerTest]
## Formula: yield ~ Fertilizer + (1 | plot/subplot)
##    Data: AgData
##
## REML criterion at convergence: 572.6
##
## Scaled residuals:
##     Min      1Q   Median      3Q      Max
## -1.78714 -0.62878 -0.08602  0.64094  2.36353
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  subplot:plot (Intercept) 5.345    2.312
##  plot        (Intercept) 8.854    2.975
##  Residual                1.014    1.007
## Number of obs: 160, groups:  subplot:plot, 32; plot, 8
##
## Fixed effects:
##               Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)    21.0211     1.2056  8.9744  17.436 3.14e-08 ***
## FertilizerHigh  4.6323     0.8328 23.0000   5.563 1.17e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##            (Intr)
## FertilzrHgh -0.345
```

Notice the standard deviation plant-to-plant noise is about 1/3 of the noise associated with subplot-to-subplot or even plot-to-plot. Finally the effect of increasing the fertilizer level is to increase yield by about 4.6.


A number of *in-situ* experiments looking at the addition CO_2 and warming on landscapes have been done (typically called Free Air CO_2 Experiments (FACE)) and these are interesting from an experimental design perspective because we have limited number of replicates because the cost of exposing plants to different CO_2 levels outside a greenhouse is extraordinarily expensive. In the `dsData` package, there is a dataset that is inspired by one of those studies.

The experimental units for the CO_2 treatment will be called a ring, and we have nine rings. We have three treatments (A,B,C) which correspond to an elevated CO_2 treatment, an ambient CO_2 treatment with all the fans, and a pure control. For each ring we'll have some measure of productivity but we have six replicate observations per ring.

```
data("HierarchicalData", package = 'dsData')
head(HierarchicalData)
```

```
##   Trt Ring Rep        y
## 1   A    1   1 363.9684
## 2   A    1   2 312.0613
## 3   A    1   3 332.9916
## 4   A    1   4 320.0109
## 5   A    1   5 292.2656
## 6   A    1   6 315.8136
```


Experiment Design and Results
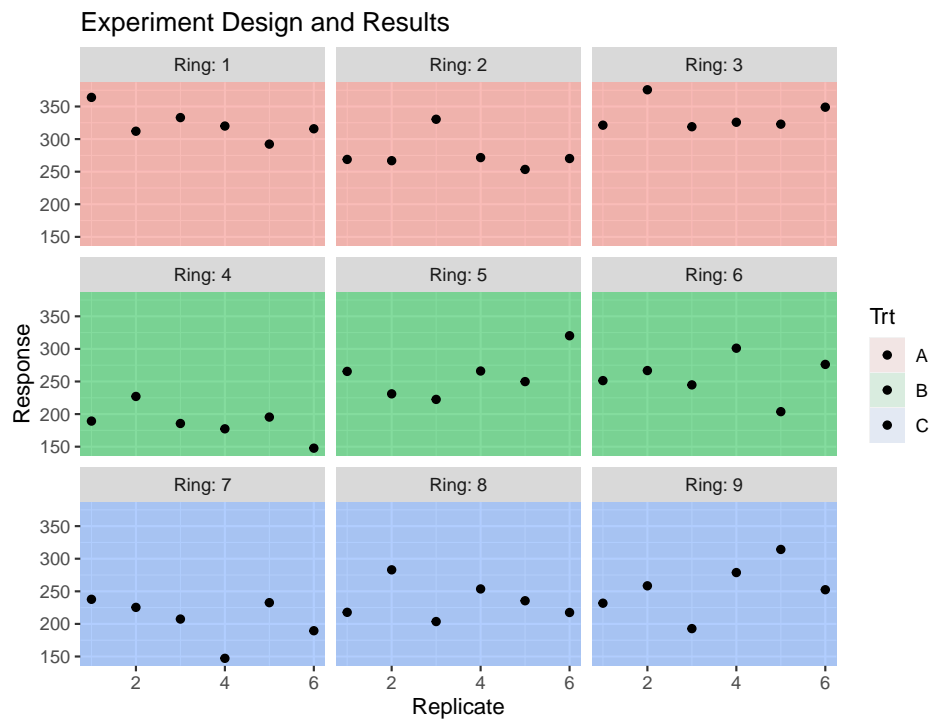
We can easily fit this model using random effects for each ring.

```
model <- lmer( y ~ Trt + (1|Ring), data=HierarchicalData )
anova(model)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##      Sum Sq Mean Sq NumDF DenDF F value  Pr(>F)
## Trt   10776  5388.1     2     6  5.7176 0.04075 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To think about what is actually going on, it is helpful to consider the predicted values from this model. As usual we will use the `predict` function, but now we have the option of including the random effects or not.

First lets consider the predicted values if we completely ignore the Ring random effect while making predictions.

```r
HierarchicalData.Predictions <- HierarchicalData %>%
  mutate( y.hat   = predict(model, re.form= ~ 0),  # don't include any random effects
          y.hat   = round( y.hat, digits=2),
          my.text = TeX(paste('$\\hat{y}$ =', y.hat), output='character')) %>%
  group_by(Trt, Ring) %>%
  slice(1)  # Predictions are the same for all replicates in the ring

HierarchicalData.Predictions %>%
  head()
```

```
## # A tibble: 6 x 6
## # Groups:   Trt, Ring [6]
##    Trt   Ring    Rep     y y.hat my.text
##    <fct> <fct> <int> <dbl> <dbl> <chr>
## 1 A     1         1  364.  312. paste('','',hat(paste('y')),'',' = 311','. 78')
## 2 A     2         1  269.  312. paste('','',hat(paste('y')),'',' = 311','. 78')
## 3 A     3         1  321.  312. paste('','',hat(paste('y')),'',' = 311','. 78')
## 4 B     4         1  189.  235. paste('','',hat(paste('y')),'',' = 234','. 53')
## 5 B     5         1  265.  235. paste('','',hat(paste('y')),'',' = 234','. 53')
## 6 B     6         1  251.  235. paste('','',hat(paste('y')),'',' = 234','. 53')
```

Treatment Only Predictions
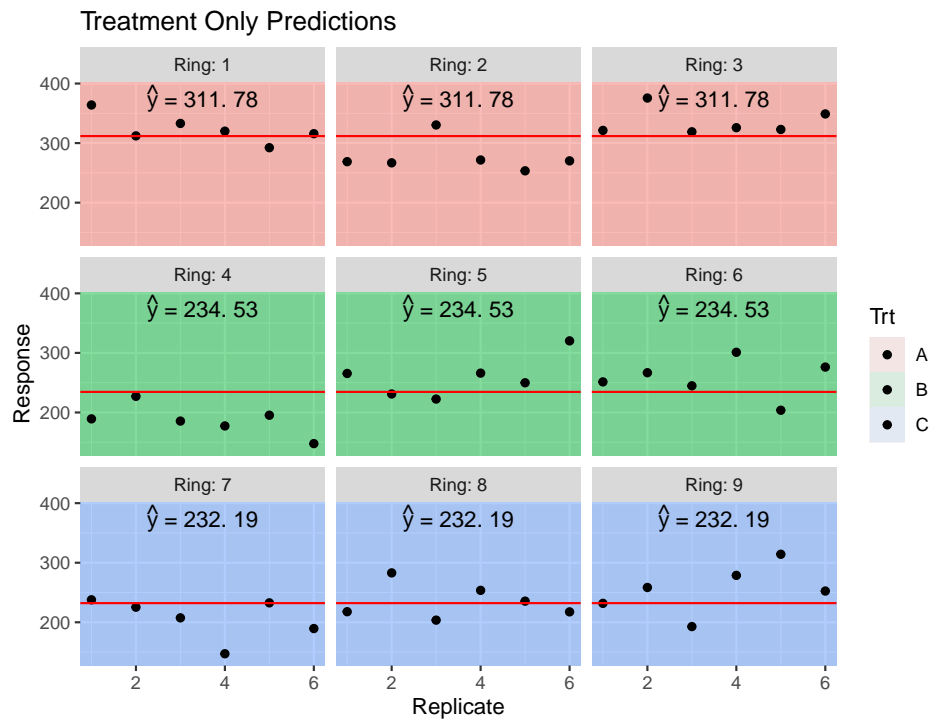
Now we consider the predicted values, but created using the Ring random effect. These random effects provide for a slight perturbation up or down depending on the quality of the Ring, but the sum of all 9 Ring effects is *required* to be 0.

```
ranef(model)
```

```
## $Ring
##   (Intercept)
## 1    9.458738
## 2  -29.798425
## 3   20.339687
## 4  -40.532972
## 5   21.067503
## 6   19.465469
## 7  -21.814405
## 8    2.548287
## 9   19.266118
##
## with conditional variances for "Ring"
```

```
sum(ranef(model)$Ring)
```

```
## [1] -3.519407e-12
```

Also notice that the sum of the random effects *within a treatment* is zero! (Recall Ring 1:3 was treatment A, 4:6 was treatment B, and 7:9 was treatment C).

```r
HierarchicalData.Predictions <- HierarchicalData %>%
  mutate( y.hat   = predict(model, re.form= ~ (1|Ring)),  # Include Ring Random effect
          y.hat   = round( y.hat, digits=2),
          my.text = TeX(paste('$\\hat{y}$ =', y.hat), output='character'))
```


Treatment and Ring Predictions
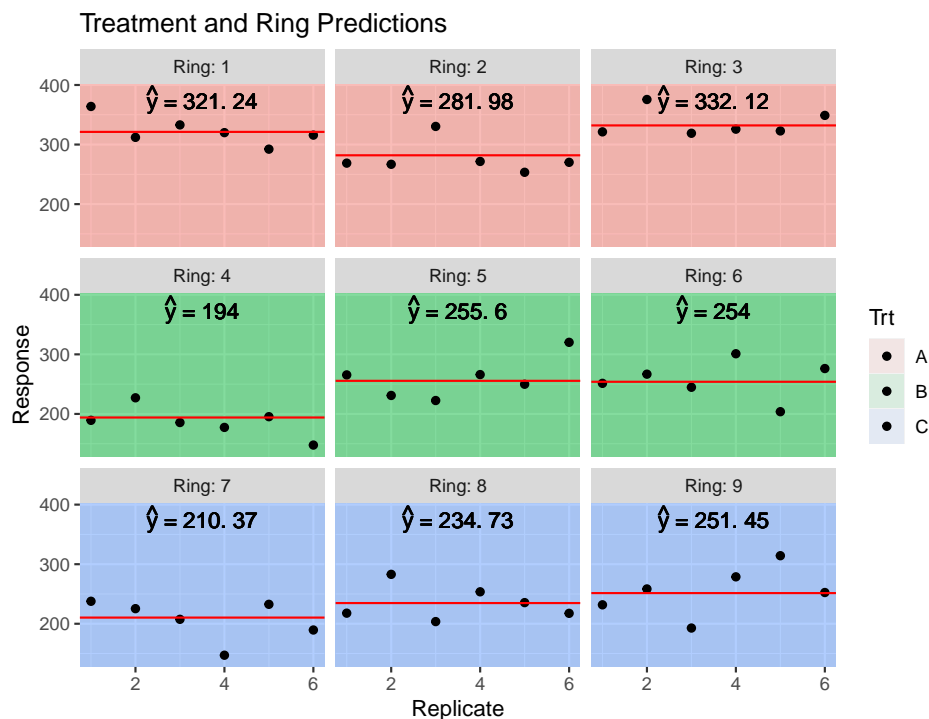
We interpret the random effect of Ring as a perturbation to expected value of the response that you expect just based on the treatment provided.

_____

We'll now consider an example with a somewhat ridiculous amount of nesting. We will consider an experiment run to test the consistency between laboratories. A large jar of dried egg power was fully homogenized and divided into a number of samples and the fat content between the samples should be the same. Six laboratories were randomly selected and instructed to have two technicians analyze what they thought were different all different samples. Each technician received two separate samples (labeled H and G) and were instructed to measure each sample twice. So our hierarchy is that observations are nested within samples which are nested within technicians which are nested in labs.

	Fat	Lab	Technician	Sample
1	0.62	I	one	G
2	0.55	I	one	G
3	0.34	I	one	H
4	0.24	I	one	H
...	...	...	...	...
45	0.18	VI	two	G
46	0.2	VI	two	G
47	0.26	VI	two	H
48	0.06	VI	two	H

Notice in the data, the technicians are always labeled as 'one' and 'two' regardless of the lab. Likewise the two samples given to each technician are always labeled 'G' and 'H' even though the actual physical samples are different for each technician.

In terms of notation, we will refer to the 6 labs as L_i and the lab technicians as T_{ij} and we note that j is either 1 or 2 which doesn't uniquely identify the technician unless we include the lab subscript as well. Finally the sub-samples are nested within the technicians and we denote them as S_{ijk}. Finally our "pure" error is the two observations from the same sample. So the model we wish to fit is:

$$y_{ijkl} = \mu + L_i + T_{ij} + S_{ijk} + \epsilon_{ijkl}$$

where $L_i \overset{iid}{\sim} N\left(0, \sigma_L^2\right)$, $T_{ij} \overset{iid}{\sim} N\left(0, \sigma_T^2\right)$, $S_{ijk} \overset{iid}{\sim} N\left(0, \sigma_S^2\right)$, $\epsilon_{ijkl} \overset{iid}{\sim} N\left(0, \sigma_\epsilon^2\right)$.

We need a convenient way to tell `lmer` which factors are nested in which. We can do this by creating data columns that make the interaction terms. For example there are 12 technicians (2 from each lab), but in our data frame we only see two levels, so to create all 12 random effects, we need to create an interaction column (or tell `lmer` to create it and use it). Likewise there are 24 sub-samples and 48 "pure" random effects.

```
data('eggs', package='faraway')
model <- lmer( Fat ~ 1 + (1|Lab) + (1|Lab:Technician) +
                     (1|Lab:Technician:Sample), data=eggs)
model <- lmer( Fat ~ 1 + (1|Lab/Technician/Sample), data=eggs)
```

```
eggs <- eggs %>%
  mutate( yhat = predict(model, re.form=~0))
ggplot(eggs, aes(x=Sample, y=Fat)) +
  geom_point() +
  geom_line(aes(y=yhat, x=as.integer(Sample)), color='red') +
  facet_grid(. ~ Lab:Technician) +
  ggtitle('Average Value Only')
```

Average Value Only
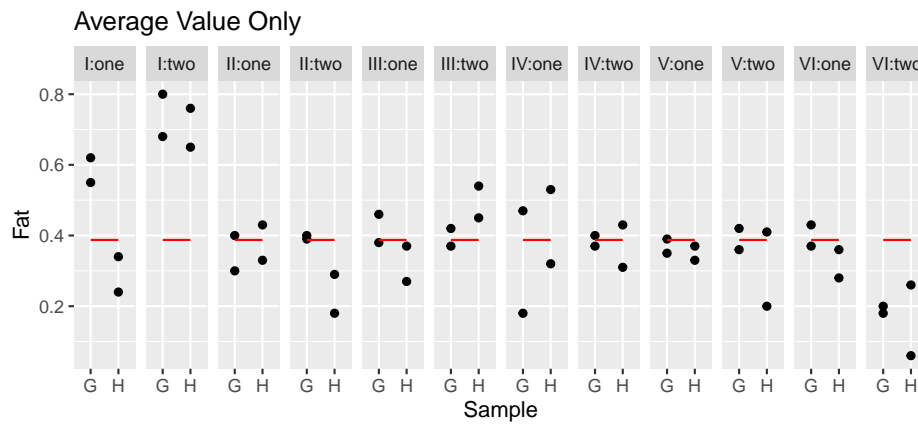

```
eggs <- eggs %>%
  mutate( yhat = predict(model, re.form=~(1|Lab)))
ggplot(eggs, aes(x=Sample, y=Fat)) +
  geom_point() +
  geom_line(aes(y=yhat, x=as.integer(Sample)), color='red') +
  facet_grid(. ~ Lab+Technician) +
  ggtitle('Average With Lab Offset')
```

Average With Lab Offset
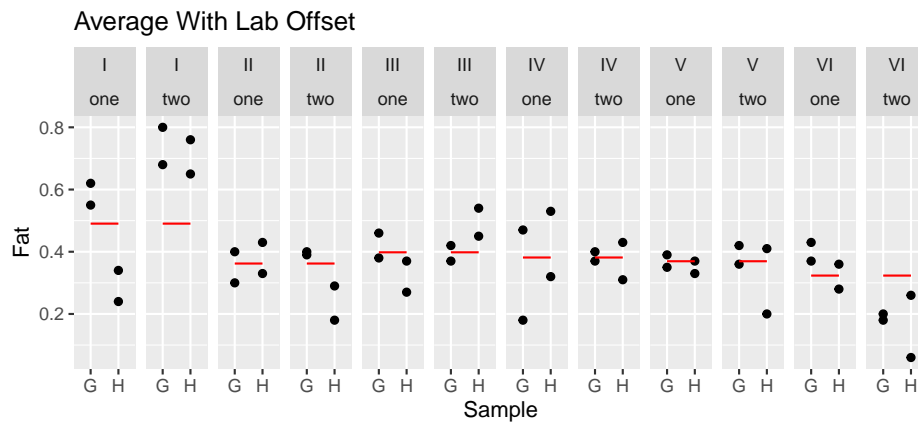

```
eggs <- eggs %>%
  mutate( yhat = predict(model, re.form=~(1|Lab/Technician)))
ggplot(eggs, aes(x=Sample, y=Fat)) +
  geom_point() +
  geom_line(aes(y=yhat, x=as.integer(Sample)), color='red') +
  facet_grid(. ~ Lab+Technician) +
  ggtitle('Average With Lab + Technician Offset')
```

Average With Lab + Technician Offset


```
eggs <- eggs %>%
  mutate( yhat = predict(model, re.form=~(1|Lab/Technician/Sample)))
ggplot(eggs, aes(x=Sample, y=Fat)) +
  geom_point() +
  geom_line(aes(y=yhat, x=as.integer(Sample)), color='red') +
  facet_grid(. ~ Lab+Technician) +
  ggtitle('Average With Lab + Technician + Sample Offset')
```
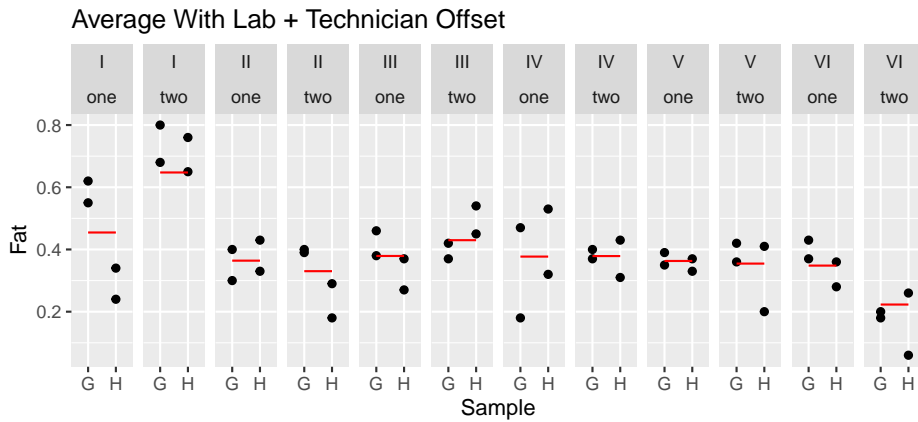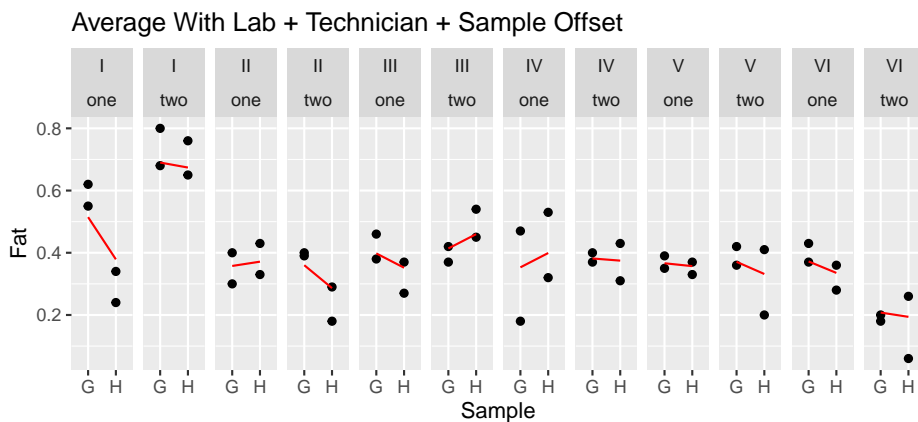
Average With Lab + Technician + Sample Offset

Now that we have an idea of how things vary, we can look at the σ terms.

```
summary(model)[['varcor']]
```

```
## Groups                Name        Std.Dev.
## Sample:(Technician:Lab) (Intercept) 0.055359
## Technician:Lab          (Intercept) 0.083548
## Lab                     (Intercept) 0.076941
## Residual                            0.084828
```

It looks like there is still plenty of unexplained variability, but the next largest source of variability is in the technician and also the lab. Is the variability lab-to-lab large enough for us to convincingly argue that it is statistically significant?

```
ranova(model)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## Fat ~ (1 | Sample:(Technician:Lab)) + (1 | Technician:Lab) +
##     (1 | Lab)
##                              npar logLik    AIC     LRT Df Pr(>Chisq)
## <none>                          5 32.118 -54.235
## (1 | Sample:(Technician:Lab))   4 31.316 -54.632 1.60342  1    0.20542
## (1 | Technician:Lab)            4 30.740 -53.480 2.75552  1    0.09692 .
## (1 | Lab)                       4 31.719 -55.439 0.79649  1    0.37215
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It looks like the technician effect is at the edge of statistically significant, but the lab-to-lab effect is smaller than the pure error and not statistically significant. I'm not thrilled with the repeatability, but the technicians are a bigger concern than the individual labs. These data aren't strong evidence of big differences between labs, but I would need to know if the size of the error is practically important before we just pick the most convenient lab to send our samples to.

10.7 Crossed Effects

If two effects are not nested, we say they are *crossed*. In the penicillin example, the treatments and blends were not nested and are therefore crossed.

An example is a Latin square experiment to look the effects of abrasion on four different material types (A, B, C, and D). We have a machine to do the abrasion test with four positions and we did 4 different machine runs. Our data looks like the following setup:

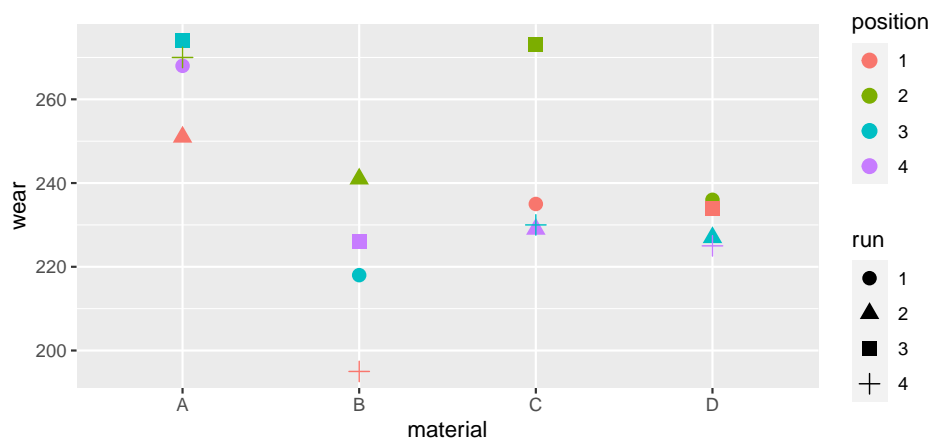| run	Position: 1	Position: 2	Position: 3	Position: 4
1	C	D	B	A
2	A	B	D	C
3	D	C	A	B
4	B	A	C	D

Our model can be written as

$$y_{ijk} = \mu + M_i + P_j + R_k + \epsilon_{ijk}$$

and we notice that the position and run effects are not nested within anything else and thus the subscript have just a single index variable. Certainly the run effect should be considered random as these four are a sample from all possible runs, but what about the position variable? Here we consider that the machine being used is a random selection from all possible abrasion machines and any position differences have likely developed over time and could be considered as a random sample of possible position effects. We'll regard both position and run as crossed random effects.

```
data('abrasion', package='faraway')
ggplot(abrasion, aes(x=material, y=wear, color=position, shape=run)) +
  geom_point(size=3)
```


It certainly looks like the materials are different. I don't think the run matters, but position 2 seems to develop excessive wear compared to the other positions.

```
m <- lmer( wear ~ material + (1|run) + (1|position), data=abrasion)
anova(m)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##          Sum Sq Mean Sq NumDF DenDF F value     Pr(>F)
## material 4621.5  1540.5     3     6  25.151 0.0008498 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The material effect is statistically significant and we can figure out the pairwise differences in the usual fashion.

```
emmeans(m, specs= pairwise~material)
```

```
## $emmeans
##  material emmean   SE   df lower.CL upper.CL
##  A           266 7.67 7.48      248      284
##  B           220 7.67 7.48      202      238
##  C           242 7.67 7.48      224      260
##  D           230 7.67 7.48      213      248
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate   SE df t.ratio p.value
##  A - B        45.8 5.53  6   8.267  0.0007
##  A - C        24.0 5.53  6   4.337  0.0190
##  A - D        35.2 5.53  6   6.370  0.0029
##  B - C       -21.8 5.53  6  -3.930  0.0295
##  B - D       -10.5 5.53  6  -1.897  0.3206
##  C - D        11.2 5.53  6   2.033  0.2743
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 4 estimates
```

```
emmeans(m, specs= ~material) %>%
  multcomp::cld(Letters=letters)
```

```
##  material emmean   SE   df lower.CL upper.CL .group
##  B           220 7.67 7.48      202      238 a
##  D           230 7.67 7.48      213      248 ab
##  C           242 7.67 7.48      224      260  b
##  A           266 7.67 7.48      248      284   c
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
```

So material D is in between materials B and C for abrasion resistance.

```
summary(m)[['varcor']]
```

```
##  Groups   Name        Std.Dev.
```

```
## run      (Intercept)  8.1790
## position (Intercept) 10.3471
## Residual              7.8262
```

Notice that run and the pure error standard deviation have about the same magnitude, but position is more substantial. Lets see what happens if we remove the run random effect.

```r
m2 <- lmer( wear ~ material + (1|position), data=abrasion) # Run is marginally signifi
# anova(m, m2)            # This would be wrong because it would refit the model by d
anova(m, m2, refit=FALSE)  # refit=FALSE keeps it using REML instead of refitting usin
```

```
## Data: abrasion
## Models:
## m2: wear ~ material + (1 | position)
## m: wear ~ material + (1 | run) + (1 | position)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m2    6 115.30 119.94 -51.651   103.30
## m     7 114.26 119.66 -50.128   100.26 3.0459  1     0.08094 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
m3 <- lmer( wear ~ material + (1|run), data=abrasion) # Position is mildly significant
anova(m, m3, refit=FALSE)  #
```

```
## Data: abrasion
## Models:
## m3: wear ~ material + (1 | run)
## m: wear ~ material + (1 | run) + (1 | position)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m3    6 116.85 121.48 -52.425   104.85
## m     7 114.26 119.66 -50.128   100.26 4.5931  1     0.0321 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Now do a similar test, but all at once.
ranova(m)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## wear ~ material + (1 | run) + (1 | position)
##                 npar  logLik    AIC    LRT Df Pr(>Chisq)
```

```
## <none>            7 -50.128 114.26
## (1 | run)         6 -51.651 115.30 3.0459  1    0.08094 .
## (1 | position)    6 -52.425 116.85 4.5931  1    0.03210 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that while position continues to be weakly statistically significant, the run has dropped down to marginal significance.

10.8   Repeated Measures / Longitudinal Studies

In repeated measurement experiments, repeated observations are taken on each subject. When those repeated measurements are taken over a sequence of time, we call it a longitudinal study. Typically covariates are also observed at the same time points and we are interested in how the response is related to the covariates.

In this case the correlation structure is that observations on the same person/object should be more similar than observations between two people/objects. As a result we need to account for repeated measures by including the person/object as a random effect.

To demonstrate a longitudinal study we turn to the data set `sleepstudy` in the `lme4` library. Eighteen patients participated in a study in which they were allowed only 3 hours of sleep per night and their reaction time in a specific test was observed. On day zero (before any sleep deprivation occurred) their reaction times were recorded and then the measurement was repeated on 9 subsequent days.

```
data('sleepstudy', package='lme4')
ggplot(sleepstudy, aes(y=Reaction, x=Days)) +
    facet_wrap(~ Subject, ncol=6) +
    geom_point() +
    geom_line()
```
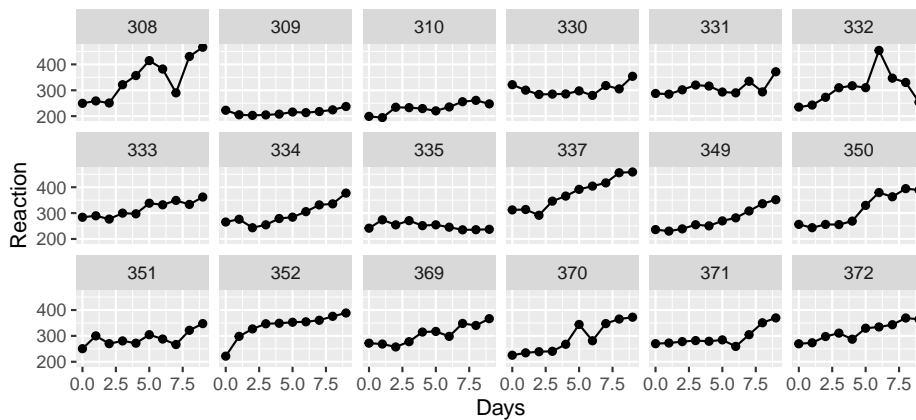
We want to fit a line to these data, but how should we do this? First we notice that each subject has their own baseline for reaction time and the subsequent measurements are relative to this, so it is clear that we should fit a model with a random intercept.

```r
m1 <- lmer( Reaction ~ Days + (1|Subject), data=sleepstudy)
summary(m1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: Reaction ~ Days + (1 | Subject)
##    Data: sleepstudy
##
## REML criterion at convergence: 1786.5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2257 -0.5529  0.0109  0.5188  4.2506
##
## Random effects:
##  Groups    Name        Variance Std.Dev.
##  Subject  (Intercept) 1378.2   37.12
##  Residual              960.5   30.99
## Number of obs: 180, groups:  Subject, 18
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 251.4051     9.7467   22.8102   25.79   <2e-16 ***
## Days         10.4673     0.8042  161.0000   13.02   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Correlation of Fixed Effects:
##      (Intr)
## Days -0.371
```

```
ranova(m1)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## Reaction ~ Days + (1 | Subject)
##                npar  logLik    AIC   LRT Df Pr(>Chisq)
## <none>            4 -893.23 1794.5
## (1 | Subject)     3 -946.83 1899.7 107.2  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To visualize how well this model fits our data, we will plot the predicted values which are lines with y-intercepts that are equal to the sum of the fixed effect of intercept and the random intercept per subject. The slope for each patient is assumed to be the same and is approximately 10.4.

```
sleepstudy <- sleepstudy %>%
  mutate(yhat = predict(m1, re.form=~(1|Subject)))
ggplot(sleepstudy, aes(y=Reaction, x=Days)) +
    facet_wrap(~ Subject, ncol=6) +
    geom_point() +
    geom_line() +
    geom_line(aes(y=yhat), color='red')
```
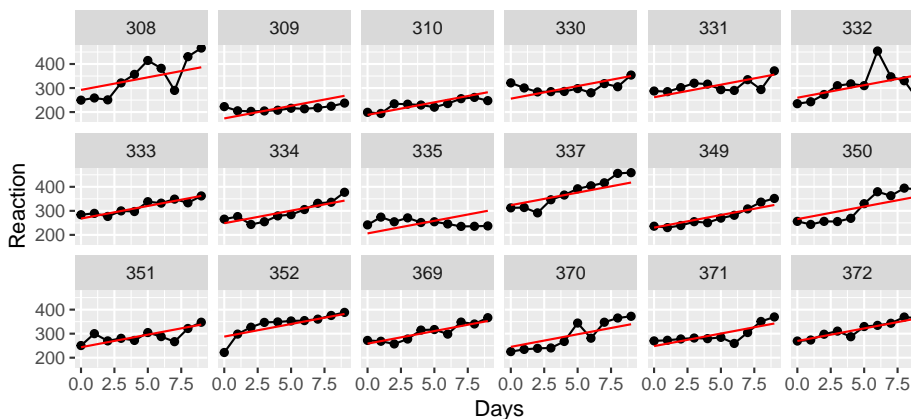

This isn't too bad, but I would really like to have each patient have their own slope as well as their own y-intercept. The random slope will be calculated as a fixed effect of slope plus a random offset from that.

```r
# Random effects for intercept and Slope
m2 <- lmer( Reaction ~ Days + ( 1+Days | Subject), data=sleepstudy)

sleepstudy <- sleepstudy %>%
  mutate(yhat = predict(m2, re.form=~(1+Days|Subject)))
ggplot(sleepstudy, aes(y=Reaction, x=Days)) +
    facet_wrap(~ Subject, ncol=6) +
    geom_point() +
    geom_line() +
    geom_line(aes(y=yhat), color='red')
```
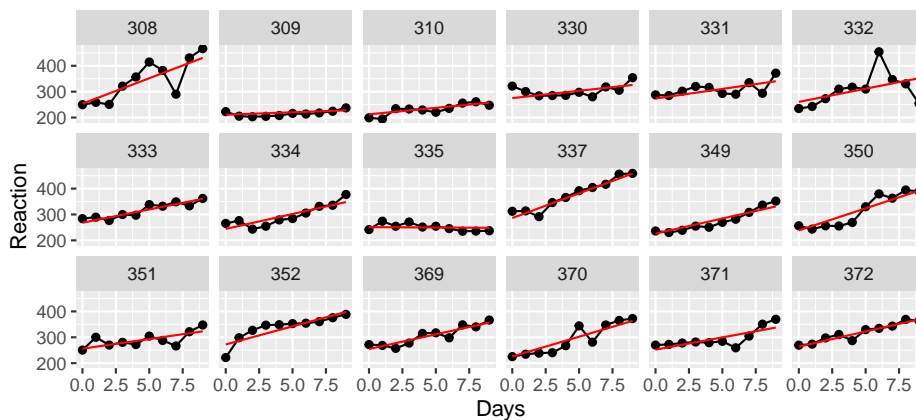

This appears to fit the observed data quite a bit better, but it is useful to test this.

```r
# This is the first time the ranova() table has considered a reduction. Here we
# consider reducing the random term from (1+Days|Subject) to (1|Subject)
ranova(m2)
```

```
## ANOVA-like table for random-effects: Single term deletions
##
## Model:
## Reaction ~ Days + (1 + Days | Subject)
##                              npar  logLik    AIC    LRT Df Pr(>Chisq)
## <none>                          6 -871.81 1755.6
## Days in (1 + Days | Subject)    4 -893.23 1794.5 42.837  2   4.99e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# We get the same analysis directly specifying which Simple/Complex model to compare
anova(m2, m1, refit=FALSE)
```

```
## Data: sleepstudy
## Models:
## m1: Reaction ~ Days + (1 | Subject)
## m2: Reaction ~ Days + (1 + Days | Subject)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m1    4 1794.5 1807.2 -893.23   1786.5
## m2    6 1755.6 1774.8 -871.81   1743.6 42.837  2   4.99e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we see that indeed the random effect for each subject in both y-intercept
and in slope is a better model that just a random offset in y-intercept.

It is instructive to look at this example from the top down. First we plot the
population regression line.

```
sleepstudy <- sleepstudy %>%
  mutate(yhat = predict(m2, re.form=~0))
ggplot(sleepstudy, aes(x=Days, y=yhat)) +
  geom_line(color='red') + ylab('Reaction') +
  ggtitle('Population Estimated Regression Curve') +
  scale_x_continuous(breaks = seq(0,9, by=2))
```
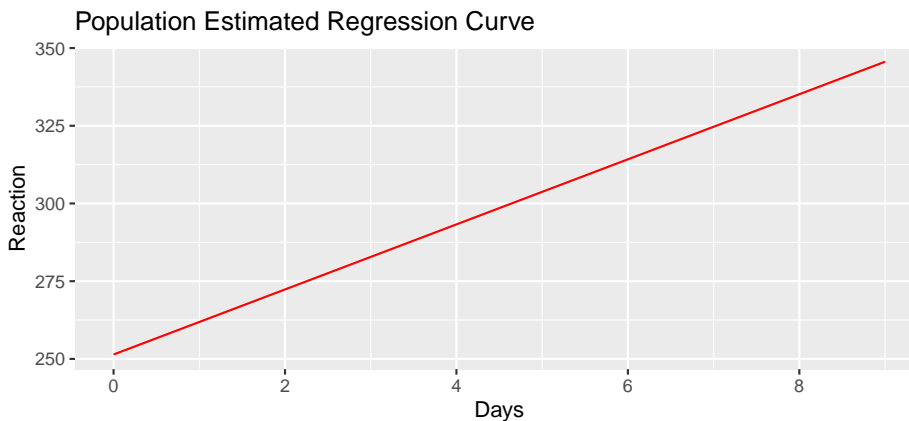


```
sleepstudy <- sleepstudy %>%
  mutate(yhat.ind = predict(m2, re.form=~(1+Days|Subject)))
ggplot(sleepstudy, aes(x=Days)) +
  geom_line(aes(y=yhat), size=3) +
  geom_line(aes(y=yhat.ind, group=Subject), color='red') +
  scale_x_continuous(breaks = seq(0,9, by=2)) +
  ylab('Reaction') + ggtitle('Person-to-Person Variation')
```

Person–to–Person Variation
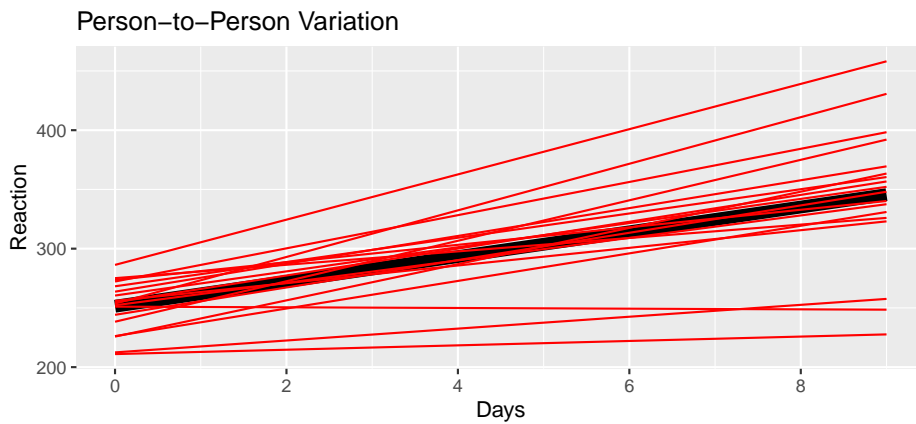

```
ggplot(sleepstudy, aes(x=Days)) +
  geom_line(aes(y=yhat)) +
  geom_line(aes(y=yhat.ind, group=Subject), color='red') +
  scale_x_continuous(breaks = seq(0,9, by=2)) +
  ylab('Reaction') + ggtitle('Within Person Variation') +
  facet_wrap(~ Subject, ncol=6) +
  geom_point(aes(y=Reaction))
```

Within Person Variation
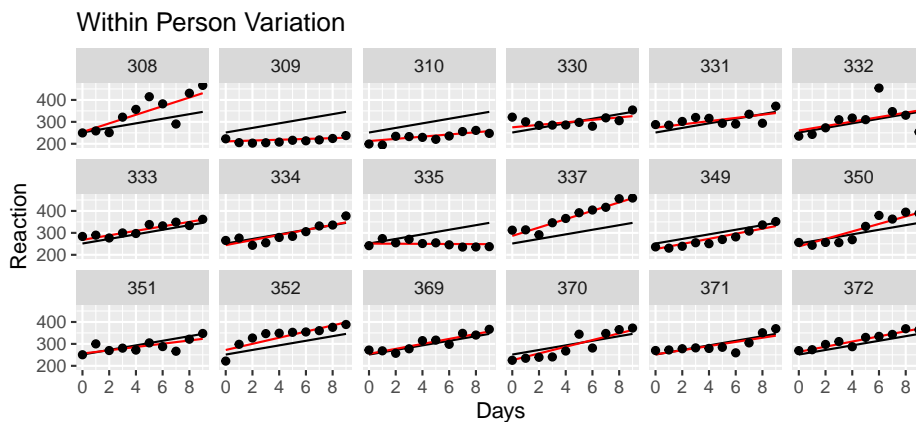
Finally we want to go back and look at the coefficients for the complex model.

```
summary(m2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: Reaction ~ Days + (1 + Days | Subject)
##    Data: sleepstudy
##
```

```
## REML criterion at convergence: 1743.6
##
## Scaled residuals:
##     Min       1Q  Median      3Q     Max
## -3.9536 -0.4634  0.0231  0.4634  5.1793
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  Subject  (Intercept) 612.10   24.741
##           Days         35.07    5.922   0.07
##  Residual             654.94   25.592
## Number of obs: 180, groups:  Subject, 18
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)  251.405      6.825  17.000  36.838  < 2e-16 ***
## Days          10.467      1.546  17.000   6.771 3.26e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##      (Intr)
## Days -0.138
```

10.9   Confidence and Prediction Intervals

As with the standard linear model, we often want to create confidence and pre-
diction intervals for a new observation or set of observations. Unfortunately,
there isn't a nice way to easily incorporate the uncertainty of the variance com-
ponents. Instead we have to rely on bootstapping techniques to produce these
quantities. Fortunately the `lme4` package provides a function that will handle
most of the looping required, but we have to describe to the program how to
create the bootstrap samples, and given a bootstrap sample, what statistics do
we want to produce intervals for.

Recall that the steps of a bootstrap procedure are:

1. Generate a bootstrap sample.
2. Fit a model to that bootstrap sample.
3. *From that model, calculate some statistic(s) you care about. This is the
   only step that the user needs do any work to specify.*
4. Repeat steps 1-3, many times, generating a bootstrap distribution of the
   statistics you care about.
5. From the bootstrap distribution generate confidence intervals for the value
   of interest.

Typically the bootstrap is used when we don't want to make any distributional assumptions on the data. In that case, we sample with replacement from the observed data to create the bootstrap data. But, if we don't mind making distributional assumptions, then instead of re-sampling the data, we could sample from the distribution with the observed parameter. In our sleep study example, we have estimated a population intercept and slope of 251.4 and 10.5. But we also have a subject intercept and slope random effect which we assumed to be normally distributed centered at zero with and with estimated standard deviations of 24.7 and 5.9. Then given a subjects regression line, observations are just normal (mean zero, standard deviation 25.6) perturbations from the line. All of these numbers came from the `summary(m2)` output.

To create a bootstrap data simulating a new subject, we could do the following:

```r
subject.intercept = 251.4 + rnorm( 1, mean = 0, sd=24.7)
subject.slope     = 10.5  + rnorm( 1, mean = 0, sd=10.5)
c(subject.intercept, subject.slope)
```

```
## [1] 276.51147  16.71648
```

```r
subject.obs <- data.frame(Days = 0:8) %>%
  mutate( Reaction = subject.intercept + subject.slope*Days + rnorm(9, sd=25.6) )

ggplot(subject.obs, aes(x=Days, y=Reaction)) + geom_point()
```
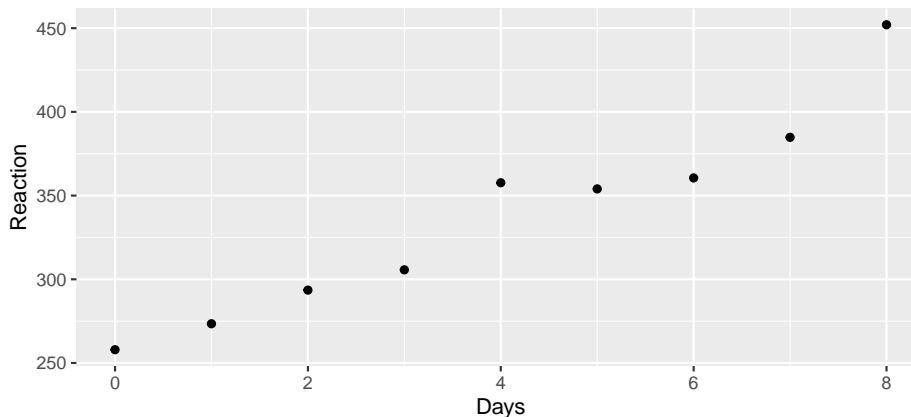

This approach is commonly referred to as a "parametric" bootstrap because we are making some assumptions about the parameter distributions, whereas in a "non-parametric" bootstrap we don't make any distributional assumptions. By default, the `bootMer` function will

1) Perform a parametric bootstrap to create new bootstrap data sets, using the results of the initial model.

2) Create a bootstrap model by analyzing the bootstrap data using the same model formula used by the initial model.
3) Apply some function you write to each bootstrap model. This function takes in a bootstrap model and returns a statistic or vector of statistics.
4) Repeat steps 1-4 repeatedly to create the bootstrap distribution of the statistics returned by your function.

10.9.1   Confidence Intervals

Now that we have a bootstrap data set, we need to take the data and then fit a model to the data and then grab the predictions from the model. At this point we are creating a confidence interval for the response line of a randomly selected person from the population. The `lme4::bootMer` function will create bootstrap data sets and then send those into the `lmer` function.

```r
ConfData <- data.frame(Days=0:8)    # What x-values I care about

# Get our best guess as to the relationship between day and reaction time
ConfData <- ConfData %>%
  mutate( Estimate = predict( m2, newdata = ConfData, re.form=~0) )

# A function to generate yhat from a model
myStats <- function(model.star){
  out <- predict( model.star, newdata=ConfData, re.form=~0 )
  return(out)
}

# bootMer generates new data sets, calls lmer on the data to produce a model,
# and then produces calls whatever function I pass in. It repeats this `nsim`
# number of times.
bootObj <- lme4::bootMer(m2, FUN=myStats, nsim = 1000 )

# using car::hist.boot, but because hist() is overloaded, I'm not able
# to call it directly. Unfortunately, it doesn't allow me to select the
# bias-corrected and accelerated method, but the percentile method is ok
# for visualizaton.
hist(bootObj, ci='perc')
```
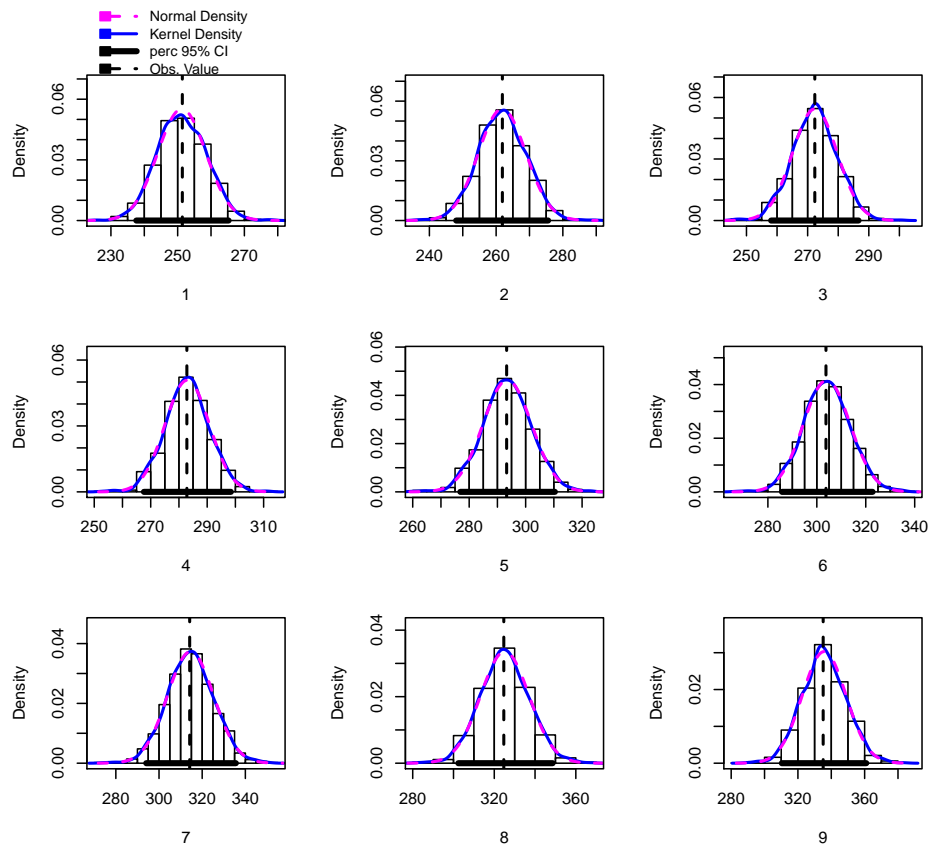
```r
# Add the confidence interval values onto estimates data frame
CI <- confint( bootObj,  level=0.95, ci='bca')
colnames(CI) <- c('lwr','upr')
ConfData <- cbind(ConfData, CI)
```

```r
# Now for a nice graph!
ConfData %>%
  ggplot(aes(x=Days)) +
  geom_line(aes(y=Estimate), color='red') +
  geom_ribbon(aes(ymin=lwr, ymax= upr), fill='salmon', alpha=0.2)
```
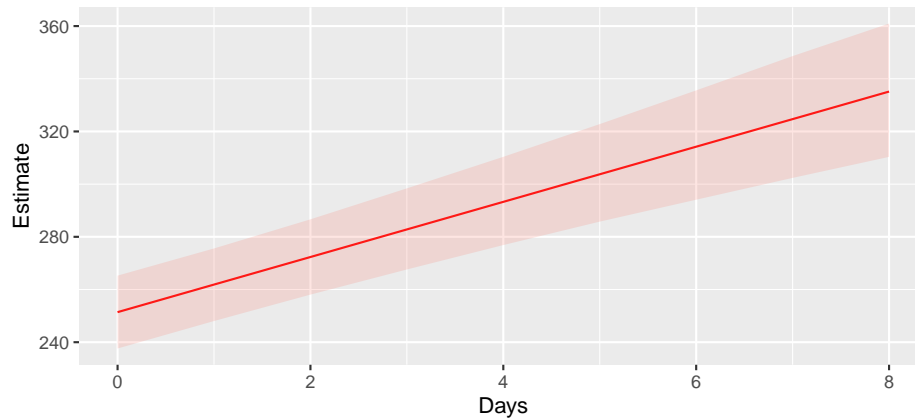
10.9.2 Prediction Intervals

For a confidence interval, we just want to find the range of observed values. In this case, we want to use the bootstrap data, but don't need to fit a model at each bootstrap step. The `lme4::simulate` function creates the bootstrap dataset and doesn't send it for more processing. It returns a vector of response values that are appropriately organized to be appended to the original dataset.

```
# # set up the structure of new subjects
PredData <- data.frame(Subject='new', Days=0:8) # Simulate a NEW patient

# Create a n x 1000 data frame
Simulated <- simulate(m2, newdata=PredData, allow.new.levels=TRUE, nsim=1000)

# squish the Subject/Day info together with the simulated and then grab the quantiles
# for each day
PredIntervals <- cbind(PredData, Simulated) %>%
  gather('sim','Reaction', sim_1:sim_1000 ) %>%    # go from wide to long structure
  group_by(Subject, Days) %>%
  summarize(lwr = quantile(Reaction, probs = 0.025),
            upr = quantile(Reaction, probs = 0.975))
```

```
## `summarise()` regrouping output by 'Subject' (override with `.groups` argument)
```

```
# Plot the prediction and confidence intervals
ggplot(ConfData,  aes(x=Days)) +
  geom_line(aes(y=Estimate), color='red') +
  geom_ribbon(aes(ymin=lwr, ymax= upr), fill='salmon', alpha=0.2) +
  geom_ribbon(data=PredIntervals, aes(ymin=lwr, ymax=upr), fill='blue', alpha=0.2)
```
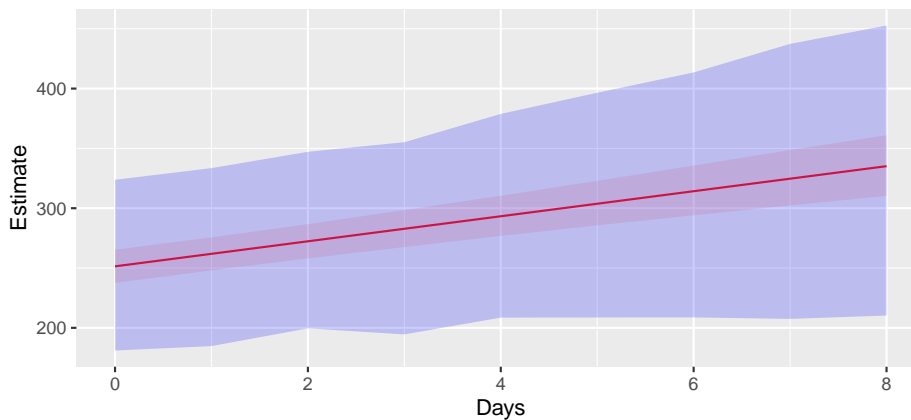
10.10   Exercises

1. An experiment was conducted to select the supplier of raw materials for production of a component. The breaking strength of the component was the objective of interest. Raw materials from four suppliers were considered. In our factory, we have four operators that can only produce one component per day. We utilized a Latin square design so that each factory operator worked with a different supplier each day. The data set is presented in the `faraway` package as `breaking`.

   a. Explain why it would be natural to treat the operators and days as random effects but the suppliers as fixed effects.
   b. Inspect the data? Does anything seem weird? It turns out that the person responsible for entering the data made an input error and transposed the response values for rows 15 and 16. Create a graph where the transposition is evident and then fix it. After your fix, make sure that each day has all 4 suppliers and 4 operators.
   c. Build a model to predict the breaking strength. Describe the variation from operator to operator and from day to day.
   d. Test the significance of the supplier effect.
   e. Is there a significant difference between the operators?

2. An experiment was performed to investigate the effect of ingestion of thyroxine or thiouracil. The researchers took 27 newborn rats and divided them into three groups. The control group is ten rats that receive no addition to their drinking water. A second group of seven rats has thyroxine added to their drinking water and the final set ten rats have thiouracil added to their water. For each of five weeks, we take a body weight measurement to monitor the rats' growth. The data are available in the `faraway` package as `ratdrink`. *I suspect that we had 30 rats to begin with*

and somehow three rats in the thyroxine group had some issue unrelated to the treatment. The following R code might be helpful for the initial visualization.

```
# we need to force ggplot to only draw lines between points for the same
# rat.  If I haven't already defined some aesthetic that is different
# for each rat, then it will connect points at the same week but for different
# rats. The solution is to add an aesthetic that does the equivalent of the
# dplyr function group_by(). In ggplot2, this aesthetic is "group".
ggplot(ratdrink, aes(y=wt, x=weeks, color=treat)) +
  geom_point(aes(shape=treat)) +
  geom_line(aes(group=subject))  # play with removing the group=subject aesthetic...
```

   a. Consider the model with an interaction between Treatment and Week along with a random effect for each subject rat. Does the model with a random offset in the y-intercept perform as well as the model with random offsets in both the y-intercept and slope?

   b. Next consider if you can simplify the model by removing the interaction between Treatment and Week and possibly even the Treatment main effect.

   c. Comment on the effect of each treatment.

3. An experiment was conducted to determine the effect of recipe and baking temperature on chocolate cake quality. For each recipe, 15 batches of cake mix for were prepared (so 45 batches total). Each batch was sufficient for six cakes. Each of the six cakes was baked at a different temperature which was randomly assigned. Several measures of cake quality were recorded of which breaking angle was just one. The dataset is available in the `faraway` package as `choccake`.

   a. For the variables Temperature, Recipe, and Batch, which should be fixed and which should be random?

   b. Inspect the data. Create a graph to consider if there is an interaction between Batch and Recipe. Describe the effect of Batch on the BreakAngle. *Hint make graphs with Batch number on the x-axis and faceting on Temperature and Recipe.*

   c. Based on the graph in part (b), it appears that there is some main effect of Batch, i.e the effect of Batch does NOT change between recipes. I would hypothesize that this is actually some "Time of Day" effect going on. (I'd love to talk the scientist who did the experiment about this). The 45 `batch:recipe` mixes need to be in the model because each batter mix resulted in 6 different cakes. Make a new variable called `mix` that has a unique value for each of the 45 `recipe:batch` levels. Then rename `batch` to be `time`. Should `time` be fixed or random?

d. Build a mixed model using the fixed effects along with the random effect of `mix`. Consider two-way interactions.

e. Using the model you selected, discuss the impact of the different variance components.

Chapter 11

Binomial Regression

```r
library(tidyverse)    # dplyr, tidyr, ggplot2, etc...
library(emmeans)
library(pROC)
library(lmerTest)
```

The standard linear model assumes that the observed data is distributed

$$y = X\beta + \epsilon \quad \text{where } \epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$$

which can be re-written as

$$y \sim N\left(\mu = X\beta, \sigma^2 I\right)$$

and notably this assumes that the data are independent. This model has $E[y] = X\beta$. This model is quite flexible and includes:

Model	Predictor Type	Response
Simple Linear Regression	1 Continuous	Continuous Normal Response
1-way ANOVA	1 Categorical	Continuous Normal Response
2-way ANOVA	2 Categorical	Continuous Normal Response
ANCOVA	1 Continuous, 1 Categorical	Continuous Normal Response

The *general* linear model expanded on the linear model and we allow the data

points to be correlated

$$y \sim N\left(X\beta, \sigma^2 \Omega\right)$$

where we assume that Ω has some known form but may include some unknown correlation parameters. This type of model includes our work with mixed models and time series data.

The study of *generalized* linear models removes the assumption that the error terms are normally distributed and allows the data to be distributed according to some other distribution such as Binomial, Poisson, or Exponential. These distributions are parameterized differently than the normal (instead of μ and σ, we might be interested in λ or p). However, I am still interested in how my covariates can be used to estimate my parameter of interest.

Critically, I still want to parameterize my covariates as $X\beta$ because we understand the how continuous and discrete covariates added and interpreted and what interactions between them mean. By keeping the $X\beta$ part, we continue to build on the earlier foundations.

11.1   Binomial Regression Model

To remove a layer of abstraction, we will now consider the case of binary regression. In this model, the observations (which we denote by w_i) are zeros and ones which correspond to some binary observation, perhaps presence/absence of an animal in a plot, or the success or failure of an viral infection. Recall that we could model this as $W_i \sim Bernoulli\left(p_i\right)$ random variable.

$$P\left(W_i = 1\right) = p_i$$

$$P\left(W_i = 0\right) = \left(1 - p_i\right)$$

which I can rewrite more formally letting w_i be the observed value as

$$P\left(W_i = w_i\right) = p_i^{w_i}\left(1 - p_i\right)^{1 - w_i}$$

and the parameter that I wish to estimate and understand is the probability of a success p_i and usually I wish to know how my covariate data $X\beta$ informs these probabilities.

In the normal distribution case, we estimated the expected value of my response vector (μ) simply using $\hat{\mu} = X\hat{\beta}$ but this will not work for an estimate of $\hat{p}$ because there is no constraint on $X\hat{\beta}$, there is nothing to prevent it from being negative or greater than 1. Because we require the probability of success to be a number between 0 and 1, I have a problem.

Example: Suppose we are interested in the abundance of mayflies in a stream. Because mayflies are sensitive to metal pollution, I might be interested in looking at the presence/absence of mayflies in a stream relative to a pollution gradient.

Here the pollution gradient is measured in Cumulative Criterion Units (CCU: CCU is defined as the ratio of the measured metal concentration to the hardness adjusted chronic criterion concentration, and then summed across each metal) where larger values imply more metal pollution.

```
data('Mayflies', package='dsData')
head(Mayflies)
```

```
##            CCU Occupancy
## 1 0.05261076         1
## 2 0.25935617         1
## 3 0.64322010         1
## 4 0.90168941         1
## 5 0.97002630         1
## 6 1.08037011         1
```

```
ggplot(Mayflies, aes(x=CCU, y=Occupancy)) + geom_point()
```


If I just fit a regular linear model to this data, we fit the following:

```
m <- lm( Occupancy ~ CCU, data=Mayflies )
Mayflies %>% mutate(yhat = predict(m)) %>%
  ggplot(aes(x=CCU, y=Occupancy)) +
  geom_point() +
  geom_line(aes(y=yhat), color='red')
```

which is horrible. First, we want the regression line to be related to the probability of occurrence and it is giving me a negative value. Instead, we want it to slowly tail off and give me more of an sigmoid-shaped curve. Perhaps something
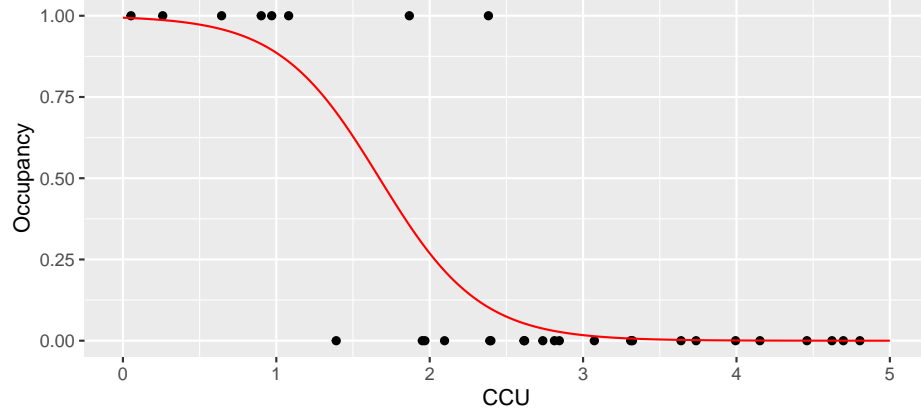
more like the following:

We need a way to convert our covariate data $y = X\beta$ from something that can take values from $-\infty$ to $+\infty$ to something that is constrained between 0 and 1 so that we can fit the model

$$w_i \sim Bernoulli\left(\underbrace{g^{-1}\left(\underbrace{y_i}_{\text{in } [-\infty, \infty]} \right)}_{\text{in } [0,1]} \right)$$

We use the notation $y_i = X_{i,\cdot}\beta$ to denote a single set of covariate values and notice that this is unconstrained and can be in $(-\infty, +\infty)$ while the parameter of interest $p_i = g^{-1}(y_i)$ is constrained to $[0, 1]$. When convenient, we will drop the i subscript while keeping the domain restrictions.

There are several options for the link function $g^{-1}(\cdot)$ that are commonly used.

1. Logit (log odds) transformation. The link function is

$$g\left(p\right) = \text{logit}(p) = \log\left[\underbrace{\frac{p}{1-p}}_{\text{odds}}\right] = y$$

with inverse

$$g^{-1}\left(y\right) = \text{ilogit}(y) = \frac{1}{1 + e^{-y}} = p$$

and we think of $g\left(p\right)$ as the log odds function.

2. Probit transformation. The link function is $g\left(p\right) = \Phi^{-1}\left(p\right)$ where Φ is the standard normal cumulative distribution function and therefore $g^{-1}\left(X\beta\right) = \Phi\left(X\beta\right)$.

3. Complementary log-log transformation: $g\left(p\right) = \log\left[-\log(1-p)\right]$.

All of these functions will give a sigmoid shape with higher probability as y increases and lower probability as it decreases. The logit and probit transformations have the nice property that if $y = 0$ then $g^{-1}\left(0\right) = \frac{1}{2}$.
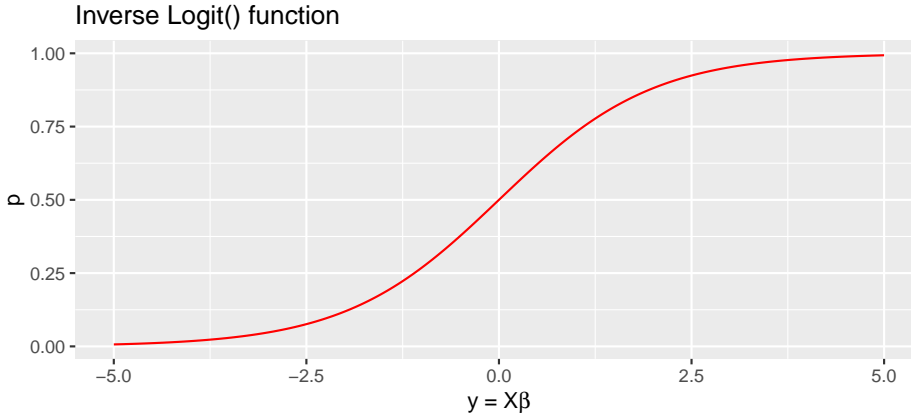
Usually the difference in inferences made using these different curves is relatively small and we will usually use the logit transformation because its form lends itself to a nice interpretation of my β values. In these cases, a slope parameter in our model will be interpreted as "the change in log odds for every one unit change in the predictor."

As in the mixed model case, there are no closed form solution for $\hat{\beta}$ and instead we must rely on numerical solutions to find the maximum likelihood estimators for $\hat{\beta}$. To do this, we must derive the likelihood function.

If

$$w_i \stackrel{iid}{\sim} Bernoulli(p_i)$$

Then

$$\mathcal{L}(p|w) = \prod_{i=1}^{n} p_i^{w_i}(1-p_i)^{1-w_i}$$

and therefore substituting $p_i = \text{ilogit}(X_i\beta)$ in we have

$$\mathcal{L}(\beta|w) = \prod_{i=1}^{n} \left(\text{ilogit}(X_i\beta)\right)^{w_i}\left(1-\text{ilogit}(X_i\beta)\right)^{1-w_i}$$

and

$$\log\mathcal{L}(\beta|w) = \sum_{i=1}^{n} \log\left\{\left(\text{ilogit}(X_i\beta)\right)^{w_i}\left(1-\text{ilogit}(X_i\beta)\right)^{1-w_i}\right\}$$

Often times it is more numerically stable to maximize the log-likelihood rather than the pure likelihood function because using logs helps prevent machine under-flow issues when the values of the likelihood is *really* small, but we will ignore that here and just assume that the function the performs the maximization is well designed to consider such issues.

We have more than one response at a particular level of X. Let m_i be the number of observations observed at the particular value of X, and w_i be the number of successes at that value of X. In that case, w_i is not a Bernoulli random variable, but rather a binomial random variable. Note that the Bernoulli distribution is the special case of the binomial distribution with $m_i = 1$.

```r
# beta are the parameters
# w is the response in terms of 0,1 for this mayfly example
# X is the design matrix that we use for our covariates.
logL <- function(beta, w, X){
  out <- dbinom( w, size=1, prob= faraway::ilogit(X%*%beta), log=TRUE ) %>% sum()
  return(out)
}
NeglogL <- function(beta, w, X){ return( -logL(beta, w, X)) }
```

```r
optim( par=c(0,0),      # initial bad guesses
       fn = NeglogL,    # we want to minimize the -logL function, maximize logL
       w=Mayflies$Occupancy,  # parameters to pass to the logL function
       X = model.matrix( ~ 1 + CCU, data=Mayflies ) )  # ditto
```

```
## $par
## [1]  5.100892 -3.050336
##
## $value
```

```
## [1] 6.324365
##
## $counts
## function gradient
##       71       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

In general, we don't want to have to specify the likelihood function by hand. Instead we will specify the model using the `glm` function which accepts a model formula as well as the distribution family that the data comes from.

```
head(Mayflies)
```

```
##          CCU Occupancy
## 1 0.05261076        1
## 2 0.25935617        1
## 3 0.64322010        1
## 4 0.90168941        1
## 5 0.97002630        1
## 6 1.08037011        1
```

```
# The following defines a "success" as if the site is
# occupied. For binomial regression, it is best to always specify the response
# as two columns where the first is the number of successes and the second is the
# number of failures for the particular set of covariates.
#
# In our case, each data row denotes a single trial and occupancy is either 0 or 1.
# To get the number of failures, we just have to subtract from the number of trials
# at the given value of CCU, which is just 1.
m1 <- glm( cbind(Occupancy, 1-Occupancy) ~ CCU, data=Mayflies, family=binomial ) #ok!
```

For binomial response data, we need to know the number of successes and the number of failures at each level of our covariate. In this case it is quite simple because there is only one observation at each CCU level, so the number of successes is Occupancy and the number of failures is just 1-Occupancy. For binomial data, `glm` expect the response to be a two-column matrix where the first column is the number successes and and the second column is the number of failures. The default choice of link function for binomial data is the logit link, but the probit can be easily chosen as well using `family=binomial(link=probit)`

in the call to `glm()`. If you only give a single response vector, it is assumed that
the second column is to be calculated as 1-first.column.

```
summary(m1)
```

```
##
## Call:
## glm(formula = cbind(Occupancy, 1 - Occupancy) ~ CCU, family = binomial,
##     data = Mayflies)
##
## Deviance Residuals:
##      Min          1Q      Median          3Q         Max
## -1.55741   -0.31594   -0.06553    0.08653     2.13362
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)     5.102      2.369    2.154   0.0313 *
## CCU            -3.051      1.211   -2.520   0.0117 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 34.795  on 29  degrees of freedom
## Residual deviance: 12.649  on 28  degrees of freedom
## AIC: 16.649
##
## Number of Fisher Scoring iterations: 7
```

Notice that the summary table includes an estimate of the standard error of
each $\hat{\beta}_j$ and a standardized value and z-test that are calculated in the usual
manner $z_j = \frac{\hat{\beta}_j - 0}{StdErr(\hat{\beta}_j)}$ but these only approximately follow a standard normal
distribution (due to the CLT results for Maximum Likelihood Estimators). We
should regard the p-values given as approximate.

The sigmoid curve shown prior was the result of the logit model and we can
estimate the probability of occupancy for any value of CCU. Surprisingly, R does
not have a built-in function for the logit and ilogit function, but the faraway
package does include them.

```
# Here are three ways to calculate the phat value for CCU = 1. The predict()
# function won't give you confidence intervals, however. So I prefer emmeans()
# new.df <- data.frame(CCU=1)
# predict(m1, newdata=new.df) %>% faraway::ilogit()  # back transform to p myself
```

```r
# predict(m1, newdata=new.df, type='response')       # ask predict() to do it
emmeans(m1, ~CCU, at=list(CCU=1), type='response')
```

```
##  CCU  prob    SE  df asymp.LCL asymp.UCL
##    1 0.886 0.129 Inf     0.391      0.99
##
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
```

```r
# Finally a nice graph
# new.df <- data.frame( CCU=seq(0,5, by=.01) )
# yhat.df <- new.df %>% mutate(fit = predict(m1, newdata=new.df, type='response') )
yhat.df <- emmeans(m1, ~CCU, at=list(CCU=seq(0,5,by=.01)), type='response') %>%
  as.data.frame()
```

```r
ggplot(Mayflies, aes(x=CCU)) +
  geom_ribbon(data=yhat.df, aes(ymin=asymp.LCL, ymax=asymp.UCL), fill='salmon', alpha=.4) +
  geom_line( data=yhat.df, aes(y=prob), color='red') +
  geom_point(aes(y=Occupancy)) +
  labs(y='Probability of Occupancy', x='Heavy Metal Pollution (CCU)', title='Mayfly Example')
```
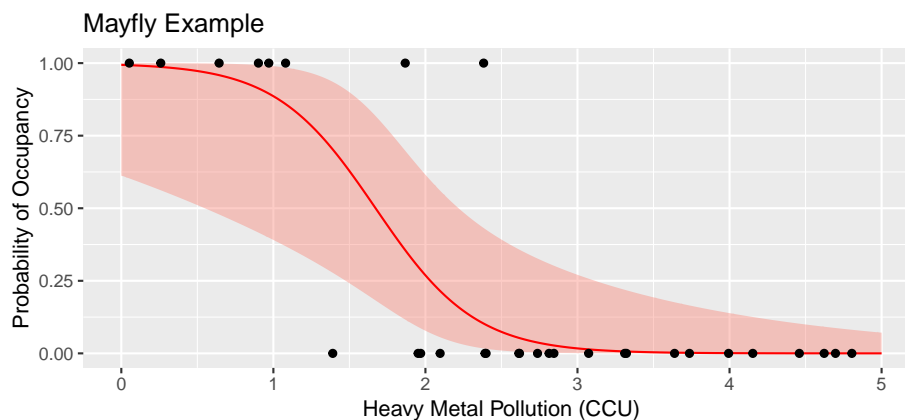

Suppose that we were to give a predicted Presence/Absence class based on the $\hat{p}$ value. Lets predict presence if the probability is greater than 0.5 and absent if the the probability is less that 0.5.

```r
Mayflies <- Mayflies %>%
  mutate(phat = predict(m1, type='response')) %>%
  mutate(chat = ifelse(phat >0.5, 1, 0))

# This is often called the "confusion matrix"
table(Truth = Mayflies$Occupancy, Prediction = Mayflies$chat)
```

```
##       Prediction
## Truth  0  1
##     0 21  1
##     1  2  6
```

This scheme has mis-classified 3 observations, two cases where mayflies were present but we predicted they would be absent, and one case where no mayflies were detected but we predicted we would see them.

11.2   Measures of Fit Quality

11.2.1   Deviance

In the normal linear models case, we were very interested in the Sum of Squared Error (SSE)

$$SSE = \sum_{i=1}^{n} (w_i - \hat{w}_i)^2$$

because it provided a mechanism for comparing the fit of two different models. If a model had a very small SSE, then it fit the observed data well. We used this as a basis for forming our F-test to compare nested models (some re-scaling by the appropriate degrees of freedom was necessary, though).

We want an equivalent measure of goodness-of-fit for models that are non-normal, but in the normal case, I would like it to be related to my SSE statistic.

The deviance of a model with respect to some data y is defined by

$$D\left(w, \hat{\theta}_0\right) = -2 \left[\log L\left(\hat{\theta}_0 | w\right) - \log L\left(\hat{\theta}_S | w\right)\right]$$

where $\hat{\theta}_0$ are the fitted parameters of the model of interest, and $\hat{\theta}_S$ are the fitted parameters under a "saturated" model that has as many parameters as it has observations and can therefore fit the data perfectly. Thus the deviance is a measure of deviation from a perfect model and is flexible enough to handle non-normal distributions appropriately.

Notice that this definition is very similar to what is calculated during the Likelihood Ratio Test. For any two models under consideration, the LRT can be formed by looking at the difference of the deviances of the two nested models

$$LRT = D\left(w, \hat{\theta}_{simple}\right) - D\left(w, \hat{\theta}_{complex}\right) \overset{\cdot}{\sim} \chi^2_{df_{complex} - df_{simple}}$$

```
m0 <- glm( Occupancy ~ 1, data=Mayflies, family=binomial )
anova(m0, m1)
```

```
## Warning in anova.glmlist(c(list(object), dotargs), dispersion = dispersion, :
## models with response '"cbind(Occupancy, 1 - Occupancy)"' removed because
## response differs from model 1
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Occupancy
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev
## NULL                   29     34.795
```

```r
1 - pchisq( 22.146, df=1 )
```

```
## [1] 2.526819e-06
```

A convenient way to get R to calculate the LRT χ^2 p-value for you is to specify the `test=LRT` inside the `anova` function.

```r
anova(m1, test='LRT')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: cbind(Occupancy, 1 - Occupancy)
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   29     34.795
## CCU   1   22.146        28     12.649 2.527e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The inference of this can be confirmed by looking at the AIC values of the two models as well.

```
AIC(m0, m1)
```

```
##    df      AIC
## m0  1 36.79491
## m1  2 16.64873
```

11.2.2   Goodness of Fit

The deviance is a good way to measure if a model fits the data, but it is not the only method. Pearson's X^2 statistic is also applicable. This statistic takes the general form $X^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$ where O_i is the number of observations observed in category i and E_i is the number expected in category i. In our case we need to figure out the categories we have. Since we have both the number of success and failures, we'll have two categories per observation i.

$$X^2 = \sum_{i=1}^{n} \left[\frac{(w_i - n_i \hat{p}_i)^2}{n_i \hat{p}_i} + \frac{((n_i - w_i) - n_i (1 - \hat{p}_i))^2}{n_i (1 - \hat{p}_i)} \right] = \sum_{i=1}^{n} \frac{(w_i - n_i \hat{p}_i)^2}{n_i \hat{p}_i (1 - \hat{p}_i)}$$

and the Pearson residual can be defined as

$$r_i = \frac{w_i - n_i \hat{p}_i}{\sqrt{n_i \hat{p}_i (1 - \hat{p}_i)}}$$

These can be found in R via the following commands

```
sum( residuals(m1, type='pearson')^2 )
```

```
## [1] 14.92367
```

Pearson's X^2 statistic is quite similar to the deviance statistic

```
deviance(m1)
```

```
## [1] 12.64873
```

11.3   Confidence Intervals

Confidence intervals for the regression could be constructed using normal approximations for the parameter estimates. An approximate $100 (1 - \alpha) \%$ confidence interval for β_i would be

$$\hat{\beta}_i \pm z^{1 - \alpha/2} \, StdErr \left(\hat{\beta}_i \right)$$

but we know that this is not a good approximation because the the normal approximation will not be good for small sample sizes and it isn't clear what is "big enough". Instead we will use an inverted LRT to develop confidence intervals for the β_i parameters.

We first consider the simplest case, where we have only an intercept and slope parameter. Below is a contour plot of the likelihood surface and the shaded region is the region of the parameter space where the parameters (β_0, β_1) would not be rejected by the LRT. This region is found by finding the maximum likelihood estimators $\hat{\beta}_0$ and $\hat{\beta}_1$, and then finding set of β_0, β_1 pairs such that

$$-2\left[\log L\left(\beta_0, \beta_1\right) - \log L\left(\hat{\beta}_0, \hat{\beta}_1\right)\right] \leq \chi^2_{df=1,0.95}$$

$$\log L\left(\beta_0, \beta_1\right) \geq \left(\frac{-1}{2}\right)\chi^2_{1,0.95} + \log L\left(\hat{\beta}_0, \hat{\beta}_1\right)$$

Log–Likelihood Surface
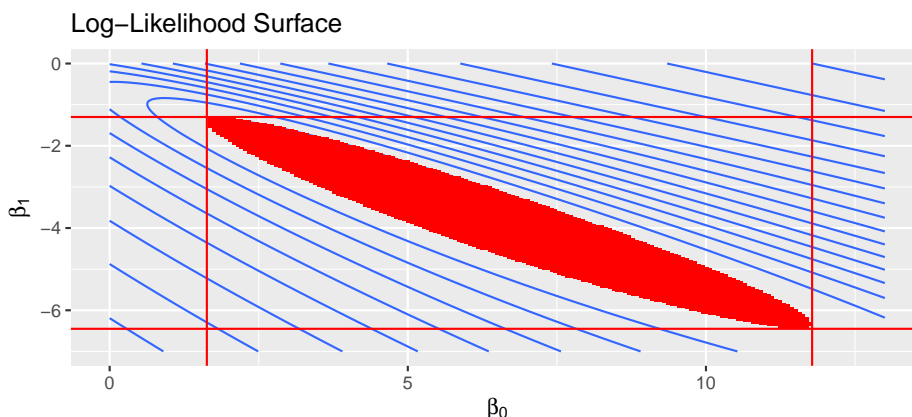
Looking at just the β_0 axis, this translates into a confidence interval of (1.63, 11.78). This method is commonly referred to as the "profile likelihood" interval because the interval is created by viewing the contour plot from the one axis. The physical analogy is to viewing a mountain range from afar and asking, "What parts of the mountain are higher than 8000 feet?"

This type of confidence interval is more robust than the normal approximation and should be used whenever practical. In R, the profile likelihood confidence interval for `glm` objects is available in the `MASS` library.

```
confint(m1) # using defaults
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %     97.5 %
## (Intercept)   1.629512  11.781167
## CCU          -6.446863  -1.304244
```

```
confint(m1, level=.95, parm='CCU') # Just the slope parameter
```

```
## Waiting for profiling to be done...
```

```
##      2.5 %    97.5 %
## -6.446863 -1.304244
```

11.4   Interpreting model coefficients

We first consider why we are dealing with odds $\frac{p}{1-p}$ instead of just p. They contain the same information, so the choice is somewhat arbitrary, however we've been using probabilities for so long that it feels unnatural to switch to odds. There are two good reasons for this, however.

The first is that the odds $\frac{p}{1-p}$ can take on any value from 0 to ∞ and so part of our translation of p to an unrestricted domain is already done.

The second is that it is easier to compare odds than to compare probabilities. For example, (as of this writing) I have a three month old baby who is prone to spitting up her milk.

- I think the probability that she will not spit up on me today is $p_1 = 0.10$. My wife disagrees and believes the probability is $p_2 = 0.01$. We can look at those probabilities and recognize that we differ in our assessment by a factor of 10 because $10 = p_1/p_2$. If we had assessed the chance of her spitting up using odds, I would have calculated $o_1 = 0.1/0.9 = 1/9$. My wife, on the other hand, would have calculated $o_2 = .01/.99 = 1/99$. The odds ratio of these is $[1/9] / [1/99] = 99/9 = 11$. This shows that she is much more certain that the event will not happen and the multiplying factor of the pair of odds is 11.

- But what if we were to consider the probability that my daughter will spit up? The probabilities assigned by me versus my wife are $p_1 = 0.9$ and $p_2 = 0.99$. How should I assess that our probabilities differ by a factor of 10, because $p_1/p_2 = 0.91 \neq 10$? The odds ratio remains the same calculation, however. The odds I would give are $o_1 = .9/.1 = 9$ vs my wife's odds $o_2 = .99/.01 = 99$. The odds ratio is now $9/99 = 1/11$ and gives the same information as I calculated from the where we defined a success as my daughter not spitting up.

To try to clear up the verbiage we'll consider a few different cases:

Probability	Odds	Verbiage
$p = .95$	$\frac{95}{5} = \frac{19}{1} = 19$	19 to 1 odds for
$p = .75$	$\frac{75}{25} = \frac{3}{1} = 3$	3 to 1 odds for
$p = .50$	$\frac{50}{50} = \frac{1}{1} = 1$	1 to 1 odds
$p = .25$	$\frac{25}{75} = \frac{1}{3} = 0.33$	3 to 1 odds against
$p = .05$	$\frac{95}{5} = \frac{1}{19} = 0.0526$	19 to 1 odds against

Given a logistic regression model with two continuous covariates, then using the `logit()` link function we have

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$\frac{p}{1-p} = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2}$$

and we can interpret β_1 and β_2 as the increase in the log odds for every unit increase in x_1 and x_2. We could alternatively interpret β_1 and β_2 using the notion that a one unit change in x_1 as a percent change of e^{β_1} in the odds. That is to say, e^{β_1} is the odds ratio of that change.

To investigate how to interpret these effects, we will consider an example of the rates of respiratory disease of babies in the first year based on covariates of gender and feeding method (breast milk, formula from a bottle, or a combination of the two). The data percentages of babies suffering respiratory disease are

	Formula f	Breast Milk b	Breast Milk + Supplement s
Males M	$\frac{77}{485}$	$\frac{47}{494}$	$\frac{19}{147}$
Females F	$\frac{48}{384}$	$\frac{31}{464}$	$\frac{16}{127}$

We can fit the saturated model (6 parameters to fit 6 different probabilities) as

```
data('babyfood', package='faraway')
head(babyfood)
```

```
##   disease nondisease  sex   food
## 1      77        381  Boy Bottle
## 2      19        128  Boy  Suppl
## 3      47        447  Boy Breast
## 4      48        336 Girl Bottle
## 5      16        111 Girl  Suppl
## 6      31        433 Girl Breast
```

```
m2 <- glm( cbind(disease,nondisease) ~ sex * food, family=binomial, data=babyfood )
summary(m2)
```

```
##
## Call:
## glm(formula = cbind(disease, nondisease) ~ sex * food, family = binomial,
##     data = babyfood)
##
## Deviance Residuals:
## [1]  0  0  0  0  0  0
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.59899    0.12495 -12.797  < 2e-16 ***
## sexGirl             -0.34692    0.19855  -1.747 0.080591 .
## foodBreast          -0.65342    0.19780  -3.303 0.000955 ***
## foodSuppl           -0.30860    0.27578  -1.119 0.263145
## sexGirl:foodBreast  -0.03742    0.31225  -0.120 0.904603
## sexGirl:foodSuppl    0.31757    0.41397   0.767 0.443012
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.6375e+01  on 5  degrees of freedom
## Residual deviance: 2.6401e-13  on 0  degrees of freedom
## AIC: 43.518
##
## Number of Fisher Scoring iterations: 3
```

Notice that the residual deviance is effectively zero with zero degrees of freedom indicating we just fit the saturated model.

It is nice to look at the single term deletions to see if the interaction term could be dropped from the model.

```
anova(m2, test='LRT')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: cbind(disease, nondisease)
##
```

```
## Terms added sequentially (first to last)
##
##
##           Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                         5    26.3753
## sex        1   5.4761         4    20.8992   0.01928 *
## food       2  20.1772         2     0.7219 4.155e-05 ***
## sex:food   2   0.7219         0     0.0000   0.69701
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Given this, we will look use the reduced model with out the interaction and check if we could reduce the model any more.

```
m1 <- glm( cbind(disease, nondisease) ~ sex + food, family=binomial, data=babyfood)
drop1(m1, test='Chi')  # all single term deletions, using LRT
```

```
## Single term deletions
##
## Model:
## cbind(disease, nondisease) ~ sex + food
##          Df Deviance    AIC     LRT  Pr(>Chi)
## <none>       0.7219 40.240
## sex       1   5.6990 43.217  4.9771   0.02569 *
## food      2  20.8992 56.417 20.1772 4.155e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this we see that we cannot reduce the model any more and we will interpret the coefficients of this model.

```
coef(m1, digits=5)  # more accuracy
```

```
## (Intercept)      sexGirl  foodBreast   foodSuppl
##  -1.6127038   -0.3125528  -0.6692946  -0.1725424
```

We interpret the intercept term as the log odds that a male child fed only formula will develop a respiratory disease in their first year. With that, we could then calculate what the probability of a male formula fed baby developing respiratory disease using following

$$-1.6127 = \log\left(\frac{p_{M,f}}{1 - p_{M,f}}\right) = \text{logit}\left(p_{M,f}\right)$$

thus

$$p_{M,f} = \text{ilogit}(-1.6127) = \frac{1}{1 + e^{1.6127}} = 0.1662$$

We notice that the odds of respiratory disease disease is

$$\frac{p_{M,f}}{1 - p_{M,f}} = \frac{0.1662}{1 - 0.1662} = 0.1993 = e^{-1.613}$$

For a female child bottle fed only formula, their probability of developing respiratory disease is

$$p_{F,f} = \frac{1}{1 + e^{-(-1.6127 - 0.3126)}} = \frac{1}{1 + e^{1.9253}} = 0.1273$$

and the associated odds are

$$\frac{p_{F,f}}{1 - p_{F,f}} = \frac{0.1273}{1 - 0.1273} = 0.1458 = e^{-1.6127 - 0.3126}$$

so we can interpret $e^{-0.3126} = 0.7315$ as the percent change in odds from male to female infants. That is to say, it is the *odds ratio* of the female infants to the males is

$$e^{-0.3126} = \frac{\left(\frac{p_{F,f}}{1 - p_{F,f}}\right)}{\left(\frac{p_{M,f}}{1 - p_{M,f}}\right)} = \frac{0.1458}{0.1993} = 0.7315$$

The interpretation here is that odds of respiratory infection for females is 73.1% than that of a similarly feed male child and I might say that being female reduces the odds of respiratory illness by 27% compared to male babies. Similarly we can calculate the change in odds ratio for the feeding types:

```
exp( coef(m1) )
```

```
## (Intercept)      sexGirl   foodBreast    foodSuppl
##   0.1993479    0.7315770    0.5120696    0.8415226
```

First we notice that the intercept term can be interpreted as the odds of infection for the reference group. The each of the offset terms are the odds ratios compared to the reference group. We see that breast milk along with formula has only 84% of the odds of respiratory disease as a formula only baby, and a breast milk fed child only has 51% of the odds for respiratory disease as the formula fed baby. We can look at confidence intervals for the odds ratios by the following:

```r
exp( confint(m1) )
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept) 0.1591988 0.2474333
## sexGirl     0.5536209 0.9629225
## foodBreast  0.3781905 0.6895181
## foodSuppl   0.5555372 1.2464312
```

We should be careful in drawing conclusions here because this study was a ret-
rospective study and the decision to breast feed a baby vs feeding with formula
is inextricably tied to socio-economic status and we should investigate if the
effect measured is due to feeding method or some other lurking variable tied to
socio-economic status.

As usual, we don't want to calculate all these quantities by hand and would
prefer if `emmeans` would do all the back-transformations for us.

```r
emmeans(m1, ~ sex+food, type='response')  # probabilities for each sex/treatment
```

```
##  sex  food    prob     SE  df asymp.LCL asymp.UCL
##  Boy  Bottle 0.1662 0.0156 Inf    0.1379    0.1990
##  Girl Bottle 0.1273 0.0143 Inf    0.1018    0.1580
##  Boy  Breast 0.0926 0.0111 Inf    0.0730    0.1168
##  Girl Breast 0.0695 0.0093 Inf    0.0533    0.0901
##  Boy  Suppl  0.1437 0.0234 Inf    0.1036    0.1958
##  Girl Suppl  0.1093 0.0194 Inf    0.0766    0.1536
##
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
```

```r
emmeans(m1, pairwise~ sex, type='response') # compare Boy / Girl
```

```
## $emmeans
##  sex    prob     SE  df asymp.LCL asymp.UCL
##  Boy  0.1309 0.0111 Inf    0.1105     0.154
##  Girl 0.0992 0.0103 Inf    0.0808     0.121
##
## Results are averaged over the levels of: food
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
##
```

```
## $contrasts
##  contrast    odds.ratio    SE  df z.ratio p.value
##  Boy / Girl        1.37 0.193 Inf 2.216   0.0267
##
## Results are averaged over the levels of: food
## Tests are performed on the log odds ratio scale
```

```
emmeans(m1, revpairwise~ sex, type='response') # compare Girl / Boy
```

```
## $emmeans
##  sex    prob     SE  df asymp.LCL asymp.UCL
##  Boy  0.1309 0.0111 Inf    0.1105     0.154
##  Girl 0.0992 0.0103 Inf    0.0808     0.121
##
## Results are averaged over the levels of: food
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
##
## $contrasts
##  contrast    odds.ratio    SE  df z.ratio p.value
##  Girl / Boy       0.732 0.103 Inf -2.216   0.0267
##
## Results are averaged over the levels of: food
## Tests are performed on the log odds ratio scale
```

```
emmeans(m1, pairwise ~ food, type='response') # compare Foods
```

```
## $emmeans
##  food     prob      SE  df asymp.LCL asymp.UCL
##  Bottle 0.1457 0.01220 Inf    0.1233    0.1712
##  Breast 0.0803 0.00877 Inf    0.0647    0.0993
##  Suppl  0.1255 0.01994 Inf    0.0913    0.1700
##
## Results are averaged over the levels of: sex
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
##
## $contrasts
##  contrast         odds.ratio    SE  df z.ratio p.value
##  Bottle / Breast       1.953 0.299 Inf  4.374  <.0001
##  Bottle / Suppl        1.188 0.244 Inf  0.839  0.6786
##  Breast / Suppl        0.609 0.132 Inf -2.296  0.0564
##
## Results are averaged over the levels of: sex
```

```
## P value adjustment: tukey method for comparing a family of 3 estimates
## Tests are performed on the log odds ratio scale
```

```
emmeans(m1, revpairwise ~ food, type='response') # compare Foods
```

```
## $emmeans
##  food     prob      SE  df asymp.LCL asymp.UCL
##  Bottle 0.1457 0.01220 Inf    0.1233    0.1712
##  Breast 0.0803 0.00877 Inf    0.0647    0.0993
##  Suppl  0.1255 0.01994 Inf    0.0913    0.1700
##
## Results are averaged over the levels of: sex
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
##
## $contrasts
##  contrast        odds.ratio     SE  df z.ratio p.value
##  Breast / Bottle      0.512 0.0783 Inf  -4.374  <.0001
##  Suppl / Bottle       0.842 0.1730 Inf  -0.839  0.6786
##  Suppl / Breast       1.643 0.3556 Inf   2.296  0.0564
##
## Results are averaged over the levels of: sex
## P value adjustment: tukey method for comparing a family of 3 estimates
## Tests are performed on the log odds ratio scale
```

11.5 Prediction and Effective Dose Levels

To demonstrate the ideas in this section, we'll use a toxicology study that examined insect mortality as a function of increasing concentrations of an insecticide.

```
data('bliss', package='faraway')
```

We first fit the logistic regression model and plot the results

```
m1 <- glm( cbind(alive, dead) ~ conc, family=binomial, data=bliss)
```

Given this, we want to develop a confidence interval for the probabilities by first calculating using the following formula. As usual, we recall that the y values live in $(-\infty, \infty)$.

$$CI_y: \quad \hat{y} \pm z^{1-\alpha/2} \, StdErr\,(\hat{y})$$

We must then convert this to the $[0, 1]$ space using the ilogit() function.

$$CI_p = \text{ilogit}\,(CI_y)$$

```
probs <- data.frame(conc=seq(0,4,by=.1))
yhat <- predict(m1, newdata=probs, se.fit=TRUE)  # list with two elements fit and se.f
yhat <- data.frame( fit=yhat$fit, se.fit = yhat$se.fit)
probs <- cbind(probs, yhat)
head(probs)
```

```
##   conc      fit     se.fit
## 1  0.0 2.323790 0.4178878
## 2  0.1 2.207600 0.4022371
## 3  0.2 2.091411 0.3868040
## 4  0.3 1.975221 0.3716158
## 5  0.4 1.859032 0.3567036
## 6  0.5 1.742842 0.3421035
```

```
probs <- probs %>% mutate(
  phat  = faraway::ilogit(fit),
  lwr   = faraway::ilogit( fit - 1.96 * se.fit ),
  upr   = faraway::ilogit( fit + 1.96 * se.fit ))
ggplot(bliss, aes(x=conc)) +
  geom_point(aes(y=alive/(alive+dead))) +
  geom_line(data=probs, aes(y=phat), color='red') +
  geom_ribbon(data=probs, aes(ymin=lwr, ymax=upr), fill='red', alpha=.3) +
  ggtitle('Bliss Insecticide Data') +
  xlab('Concentration') + ylab('Proportion Alive')
```
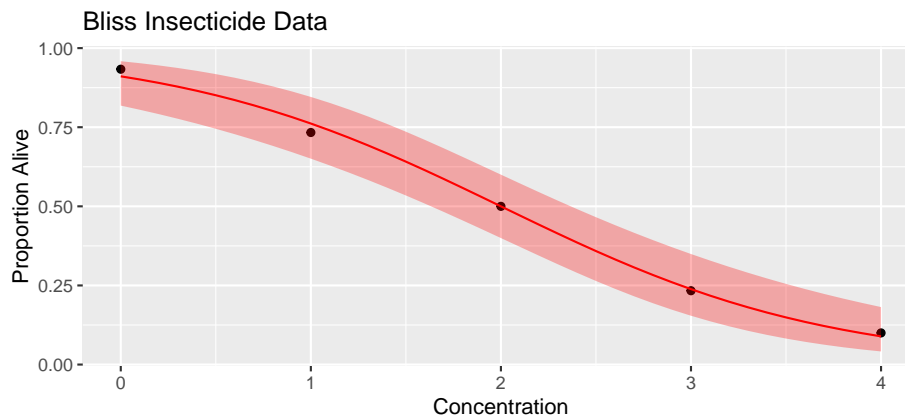

Alternatively, we might want to do this calculation via `emmeans`.

```
emmeans::emmeans(m1, ~conc, at=list(conc=c(0:4)), type='response')
```

```
## conc   prob    SE  df asymp.LCL asymp.UCL
```

```
##      0 0.9108 0.0339 Inf    0.8183    0.959
##      1 0.7617 0.0500 Inf    0.6507    0.846
##      2 0.5000 0.0518 Inf    0.3998    0.600
##      3 0.2383 0.0500 Inf    0.1542    0.349
##      4 0.0892 0.0339 Inf    0.0414    0.182
##
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
```

The next thing we want to do is come up with a confidence intervals for the concentration level that results in the death of $100(p)\%$ of the insects. Often we are interested in the case of $p = 0.5$. This is often called LD50, which is the lethal dose for 50% of the population. Using the link function you can set the p value and solve for the concentration value to find

$$\hat{x}_p = \frac{\text{logit}(p) - \hat{\beta}_0}{\hat{\beta}_1}$$

which gives us a point estimate of LD(p). To get a confidence interval we need to find the standard error of $\hat{x}_p$. Since this is a non-linear function of $\hat{\beta}_0$ and $\hat{\beta}_1$ which are correlated, we must be careful in the calculation. The actual calculation is done using the Delta Method Approximation:

$$Var\left(g\left(\hat{\theta}\right)\right) = g'\left(\theta\right)^T Var\left(\theta\right) g'\left(\theta\right)$$

Fortunately we don't have to do these calculations by hand and can use the `dose.p()` function in the `MASS` package.

```
LD <- MASS::dose.p(m1, p=c(.25, .5, .75))
LD
```

```
##                  Dose        SE
## p = 0.25: 2.945535 0.2315932
## p = 0.50: 2.000000 0.1784367
## p = 0.75: 1.054465 0.2315932
```

and we can use these to create approximately confidence intervals for these $\hat{x}_p$ values via

$$\hat{x}_p \pm z^{1-\alpha/2} StdErr\left(\hat{x}_p\right)$$

```
# why did the MASS authors make LD a vector of the
# estimated values and have an additional attribute
# that contains the standard errors?  Whatever, lets
# turn this into a convential data.frame.
str(LD)
```

```
##  'glm.dose' Named num [1:3] 2.95 2 1.05
##  - attr(*, "names")= chr [1:3] "p = 0.25:" "p = 0.50:" "p = 0.75:"
##  - attr(*, "SE")= num [1:3, 1] 0.232 0.178 0.232
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "p = 0.25:" "p = 0.50:" "p = 0.75:"
##   .. ..$ : NULL
##  - attr(*, "p")= num [1:3] 0.25 0.5 0.75
```

```r
CI <- data.frame(p = attr(LD,'p'),
                 Dose = as.vector(LD),
                 SE   = attr(LD,'SE')) %>%  # save the output table as LD
  mutate( lwr = Dose - qnorm(.975)*SE,
          upr = Dose + qnorm(.975)*SE )
CI
```

```
##      p      Dose         SE        lwr       upr
## 1 0.25 2.945535 0.2315932 2.4916207 3.399449
## 2 0.50 2.000000 0.1784367 1.6502705 2.349730
## 3 0.75 1.054465 0.2315932 0.6005508 1.508379
```

11.6   Overdispersion

In the binomial distribution, the variance is a function of the probability of success and is

$$Var\left(W\right) = np\left(1-p\right)$$

but there are many cases where we might be interested in adding an additional variance parameter ϕ to the model. A common reason for overdispersion to appear is that we might not have captured all the covariates that influence p.

We can do a quick simulation to demonstrate that additional variability in p leads to addition variability overall.

```r
N <- 1000
n <- 10
p <- .6
overdispersed_p <- p + rnorm(n, mean=0, sd=.05)
sim.data <- NULL
for( i in 1:N ){
  sim.data <- sim.data %>% rbind(data.frame(
    var = var( rbinom(N, size=n, prob=p)),
    type = 'Standard'))
  sim.data <- sim.data %>% rbind(data.frame(
    var = var( rbinom(N, size=n, prob=overdispersed_p )),
```

```
      type = 'OverDispersed'))
}
true.var <- p*(1-p)*n
ggplot(sim.data, aes(x=var, y=..density..)) +
  geom_histogram(bins=30) +
  geom_vline(xintercept = true.var, color='red') +
  facet_grid(type~.) +
  ggtitle('Histogram of Sample Variances')
```


Histogram of Sample Variances
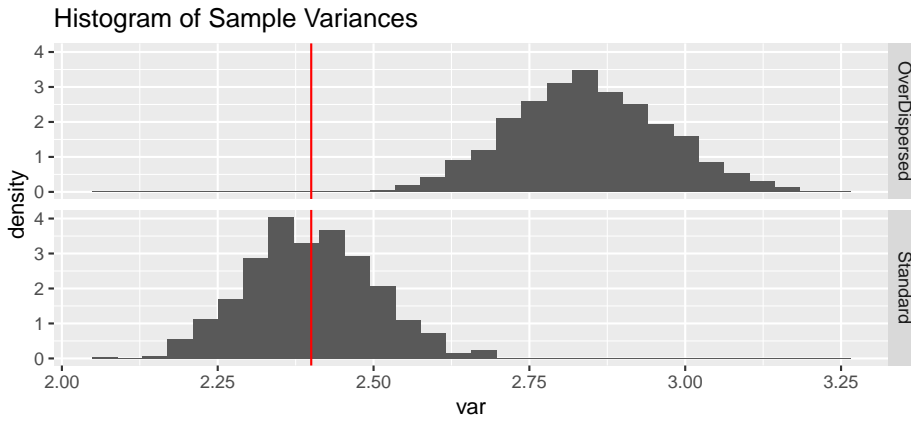
We see that the sample variances fall neatly about the true variance of 2.4 in the case where the data is distributed with a constant value for p. However adding a small amount of random noise about the parameter p, and we'd have more variance in the samples.

The extra uncertainty of the probability of success results in extra variability in the responses.

We can recognize when overdispersion is present by examining the deviance of our model. Because the deviance is approximately distributed

$$D\left(y, \theta\right) \overset{\cdot}{\sim} \chi^2_{df}$$

where df is the residual degrees of freedom in the model. Because the χ^2_k is the sum of k independent, squared standard normal random variables, it has an expectation k and variance $2k$. For binomial data with group sizes (say larger than 5), this approximation isn't too bad and we can detect overdispersion. For binary responses, the approximation is quite poor and we cannot detect overdispersion.

The simplest approach for modeling overdispersion is to introduce an addition dispersion parameter σ^2. This dispersion parameter may be estimated using

$$\hat{\sigma}^2 = \frac{X^2}{n - p}.$$

With the addition of the overdispersion parameter to the model, the differences between a simple and complex model is no longer distributed χ^2 and we must use the following approximate F-statistic

$$F = \frac{\left(D_{simple} - D_{complex}\right) / \left(df_{small} - df_{large}\right)}{\widehat{\sigma}^2}$$

Using the F-test when the the overdispersion parameter is 1 is a less powerful test than the χ^2 test, so we'll only use the F-test when the overdispersion parameter must be estimated.

Example: We consider an experiment where at five different stream locations, four boxes of trout eggs were buried and retrieved at four different times after the original placement. The number of surviving eggs was recorded and the eggs disposed of.

```
data(troutegg, package='faraway')
troutegg <- troutegg %>%
  mutate( perish = total - survive) %>%
  dplyr::select(location, period, survive, perish, total) %>%
  arrange(location, period)

troutegg %>% arrange(location, period)
```

```
##    location period survive perish total
## 1         1      4      89      5    94
## 2         1      7      94      4    98
## 3         1      8      77      9    86
## 4         1     11     141     14   155
## 5         2      4     106      2   108
## 6         2      7      91     15   106
## 7         2      8      87      9    96
## 8         2     11     104     18   122
## 9         3      4     119      4   123
## 10        3      7     100     30   130
## 11        3      8      88     31   119
## 12        3     11      91     34   125
## 13        4      4     104      0   104
## 14        4      7      80     17    97
## 15        4      8      67     32    99
## 16        4     11     111     21   132
## 17        5      4      49     44    93
## 18        5      7      11    102   113
## 19        5      8      18     70    88
## 20        5     11       0    138   138
```

We can first visualize the data

```
ggplot(troutegg, aes(x=period, y=survive/total, color=location)) +
  geom_point(aes(size=total)) +
  geom_line(aes(x=as.integer(period)))
```
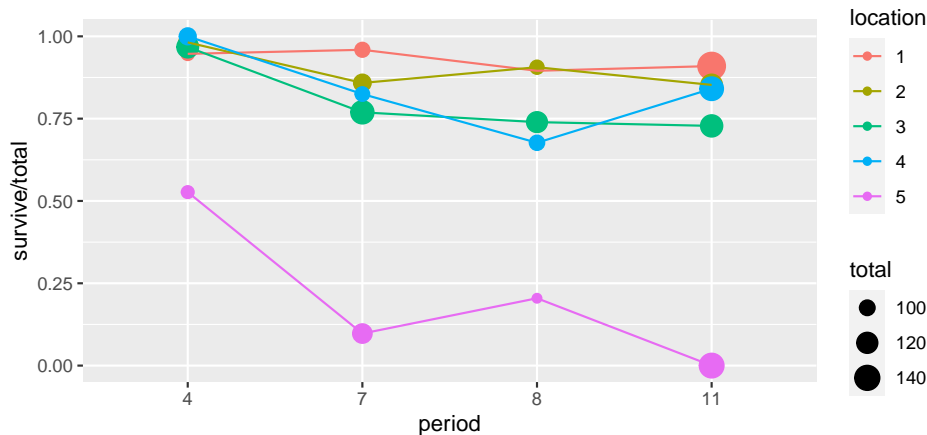

We can fit the logistic regression model (noting that the model with the inter-
action of location and period would be saturated):

```
m <- glm(cbind(survive,perish) ~ location * period, family=binomial, data=troutegg)
summary(m)
```

```
##
## Call:
## glm(formula = cbind(survive, perish) ~ location * period, family = binomial,
##     data = troutegg)
##
## Deviance Residuals:
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)         2.8792     0.4596   6.265 3.74e-10 ***
## location2           1.0911     0.8489   1.285  0.19870
## location3           0.5136     0.6853   0.749  0.45356
## location4          24.4707 51676.2179   0.000  0.99962
## location5          -2.7716     0.5044  -5.495 3.90e-08 ***
## period7             0.2778     0.6869   0.404  0.68591
## period8            -0.7326     0.5791  -1.265  0.20582
## period11           -0.5695     0.5383  -1.058  0.29007
## location2:period7  -2.4453     1.0291  -2.376  0.01749 *
```

```
## location3:period7      -2.4667      0.8796  -2.804  0.00504 **
## location4:period7     -26.0789 51676.2179  -0.001  0.99960
## location5:period7      -2.6125      0.7847  -3.329  0.00087 ***
## location2:period8      -0.9690      0.9836  -0.985  0.32453
## location3:period8      -1.6169      0.7983  -2.025  0.04284 *
## location4:period8     -25.8783 51676.2179  -0.001  0.99960
## location5:period8      -0.7331      0.6696  -1.095  0.27354
## location2:period11     -1.6468      0.9297  -1.771  0.07651 .
## location3:period11     -1.8388      0.7672  -2.397  0.01654 *
## location4:period11    -25.1154 51676.2179   0.000  0.99961
## location5:period11    -27.1679 51597.7368  -0.001  0.99958
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.0215e+03  on 19  degrees of freedom
## Residual deviance: 5.5198e-10  on  0  degrees of freedom
## AIC: 116.53
##
## Number of Fisher Scoring iterations: 22
```

The residual deviance seems a little large. With 12 residual degrees of freedom, the deviance should be near 12. We can confirm that the deviance is quite large via:

```
1 - pchisq( 64.5, df=12 )
```

```
## [1] 3.372415e-09
```

We therefore estimate the overdispersion parameter

```
sigma2 <- sum( residuals(m, type='pearson') ^2 ) / 12
sigma2
```

```
## [1] 2.29949e-11
```

and note that this is quite a bit larger than 1, which is what it should be in the non-overdispersed setting. Using this we can now test the significance of the effects of location and period.

```
drop1(m, scale=sigma2, test='F')
```

```
## Warning in drop1.glm(m, scale = sigma2, test = "F"): F test assumes
## 'quasibinomial' family
```

```
## Warning in pf(q = q, df1 = df1, ...): NaNs produced
```

```
## Single term deletions
##
## Model:
## cbind(survive, perish) ~ location * period
##
## scale:  2.29949e-11
##
##                 Df Deviance        AIC F value Pr(>F)
## <none>                  0.000 1.1700e+02
## location:period 12   64.495 2.8048e+12         0
```

and conclude that both location and period are significant predictors of trout egg survivorship.

We could have avoided having to calculate $\hat{\sigma}^2$ by hand by simply using the `quasibinomial` family instead of the binomial.

```
m2 <- glm(cbind(survive,perish) ~ location + period,
          family=quasibinomial, data=troutegg)
summary(m2)
```

```
##
## Call:
## glm(formula = cbind(survive, perish) ~ location + period, family = quasibinomial,
##     data = troutegg)
##
## Deviance Residuals:
##     Min      1Q   Median      3Q      Max
## -4.8305  -0.3650  -0.0303   0.6191   3.2434
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.6358     0.6495    7.138 1.18e-05 ***
## location2    -0.4168     0.5682   -0.734 0.477315
## location3    -1.2421     0.5066   -2.452 0.030501 *
## location4    -0.9509     0.5281   -1.800 0.096970 .
## location5    -4.6138     0.5777   -7.987 3.82e-06 ***
## period7      -2.1702     0.5504   -3.943 0.001953 **
## period8      -2.3256     0.5609   -4.146 0.001356 **
## period11     -2.4500     0.5405   -4.533 0.000686 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 5.330358)
##
##     Null deviance: 1021.469  on 19  degrees of freedom
## Residual deviance:   64.495  on 12  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```
drop1(m2, test='F')
```

```
## Single term deletions
##
## Model:
## cbind(survive, perish) ~ location + period
##          Df Deviance F value     Pr(>F)
## <none>         64.50
## location  4   913.56  39.494 8.142e-07 ***
## period    3   228.57  10.176  0.001288 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# anova(m2, test='F')
```

While each of the time periods is different than the first, it looks like periods 7,8, and 11 aren't different from each other. As usual, we need to turn to the emmeans package for looking at the pairwise differences between the periods.

```
emmeans(m2, pairwise~period, type='response')
```

```
## $emmeans
##  period prob     SE  df asymp.LCL asymp.UCL
##  4      0.960 0.0177 Inf     0.907     0.984
##  7      0.735 0.0567 Inf     0.611     0.831
##  8      0.704 0.0618 Inf     0.571     0.809
##  11     0.677 0.0549 Inf     0.562     0.774
##
## Results are averaged over the levels of: location
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
##
## $contrasts
```

```
##   contrast odds.ratio    SE  df z.ratio p.value
##   4 / 7            8.76 4.821 Inf 3.943   0.0005
##   4 / 8           10.23 5.740 Inf 4.146   0.0002
##   4 / 11          11.59 6.263 Inf 4.533   <.0001
##   7 / 8            1.17 0.475 Inf 0.383   0.9810
##   7 / 11           1.32 0.501 Inf 0.739   0.8814
##   8 / 11           1.13 0.431 Inf 0.327   0.9880
##
## Results are averaged over the levels of: location
## P value adjustment: tukey method for comparing a family of 4 estimates
## Tests are performed on the log odds ratio scale
```

```r
emmeans(m2, ~period, type='response') %>% multcomp::cld(Letters=letters)
```

```
##   period  prob     SE  df asymp.LCL asymp.UCL .group
##   11     0.677 0.0549 Inf     0.562     0.774  a
##   8      0.704 0.0618 Inf     0.571     0.809  a
##   7      0.735 0.0567 Inf     0.611     0.831  a
##   4      0.960 0.0177 Inf     0.907     0.984   b
##
## Results are averaged over the levels of: location
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
## P value adjustment: tukey method for comparing a family of 4 estimates
## Tests are performed on the log odds ratio scale
## significance level used: alpha = 0.05
```

Looking at this experiment, it looks like there is an interaction between location
and period. If we fit a model with the interaction, it is the saturated model (20
covariates for 20 observations)

```r
m3 <- glm( cbind(survive, perish) ~ period * location, family=binomial, data=troutegg)
summary(m3)
```

```
##
## Call:
## glm(formula = cbind(survive, perish) ~ period * location, family = binomial,
##     data = troutegg)
##
## Deviance Residuals:
##  [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)              2.8792     0.4596   6.265 3.74e-10 ***
## period7                  0.2778     0.6869   0.404  0.68591
## period8                 -0.7326     0.5791  -1.265  0.20582
## period11                -0.5695     0.5383  -1.058  0.29007
## location2                1.0911     0.8489   1.285  0.19870
## location3                0.5136     0.6853   0.749  0.45356
## location4               24.4707 51676.2184   0.000  0.99962
## location5               -2.7716     0.5044  -5.495 3.90e-08 ***
## period7:location2       -2.4453     1.0291  -2.376  0.01749 *
## period8:location2       -0.9690     0.9836  -0.985  0.32453
## period11:location2      -1.6468     0.9297  -1.771  0.07651 .
## period7:location3       -2.4667     0.8796  -2.804  0.00504 **
## period8:location3       -1.6169     0.7983  -2.025  0.04284 *
## period11:location3      -1.8388     0.7672  -2.397  0.01654 *
## period7:location4      -26.0789 51676.2184  -0.001  0.99960
## period8:location4      -25.8783 51676.2184  -0.001  0.99960
## period11:location4     -25.1154 51676.2184   0.000  0.99961
## period7:location5       -2.6125     0.7847  -3.329  0.00087 ***
## period8:location5       -0.7331     0.6696  -1.095  0.27354
## period11:location5     -27.1679 51597.7368  -0.001  0.99958
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.0215e+03  on 19  degrees of freedom
## Residual deviance: 5.5196e-10  on  0  degrees of freedom
## AIC: 116.53
##
## Number of Fisher Scoring iterations: 22
```

Notice that we've fit the saturated model and we've resolved the overdispersion problem because of that.

As usual, we can now look at the effect of `period` at each of the locations.

```
emmeans(m3, ~ period | location, type='response') %>% multcomp::cld()
```

```
## location = 1:
##  period      prob         SE  df asymp.LCL asymp.UCL .group
##  8       0.8953488 0.03300798 Inf 0.8109406 0.9446443  1
##  11      0.9096774 0.02302375 Inf 0.8532710 0.9457777  1
##  4       0.9468085 0.02314665 Inf 0.8785095 0.9776867  1
##  7       0.9591837 0.01998733 Inf 0.8962639 0.9845962  1
##
```

```
## location = 2:
##  period       prob          SE  df asymp.LCL asymp.UCL .group
##  11      0.8524590 0.03210799 Inf 0.7779341 0.9050269  1
##  7       0.8584906 0.03385381 Inf 0.7784456 0.9128537  1
##  8       0.9062500 0.02974911 Inf 0.8295443 0.9504977  12
##  4       0.9814815 0.01297276 Inf 0.9289964 0.9953638   2
##
## location = 3:
##  period       prob          SE  df asymp.LCL asymp.UCL .group
##  11      0.7280000 0.03980111 Inf 0.6434907 0.7987421  1
##  8       0.7394958 0.04023479 Inf 0.6533948 0.8104142  1
##  7       0.7692308 0.03695265 Inf 0.6891126 0.8336850  1
##  4       0.9674797 0.01599358 Inf 0.9165610 0.9877408   2
##
## location = 4:
##  period       prob          SE  df asymp.LCL asymp.UCL .group
##  8       0.6767677 0.04700674 Inf 0.5787856 0.7613551  1
##  7       0.8247423 0.03860215 Inf 0.7360186 0.8881766  12
##  11      0.8409091 0.03183558 Inf 0.7682755 0.8939194   2
##  4       1.0000000 0.00000007 Inf 0.0000000 1.0000000  12
##
## location = 5:
##  period       prob          SE  df asymp.LCL asymp.UCL .group
##  11      0.0000000 0.00000005 Inf 0.0000000 1.0000000  12
##  7       0.0973451 0.02788552 Inf 0.0547290 0.1672733  1
##  8       0.2045455 0.04299929 Inf 0.1328383 0.3015023  1
##  4       0.5268817 0.05177260 Inf 0.4256954 0.6259069   2
##
## Confidence level used: 0.95
## Intervals are back-transformed from the logit scale
## P value adjustment: tukey method for comparing a family of 4 estimates
## Tests are performed on the log odds ratio scale
## significance level used: alpha = 0.05
```

Notice that there isn't a huge difference in period for locations 1-4, but in location 5 things are very different.

We might consider that location really ought to be a random effect. Fortunately lme4 supports the family option, although it will not accept quasi families, you either fit a random effect or fit a quasibinomial. In either case, fitting the full interaction model with `period*location` doesn't work because we have a saturated model

```
# addititive model
m3 <- glmer(cbind(survive,perish) ~ period + (1|location),
         family=binomial, data=troutegg)
```

```
summary(m3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: cbind(survive, perish) ~ period + (1 | location)
##    Data: troutegg
##
##      AIC      BIC   logLik deviance df.resid
##    180.3    185.3    -85.2    170.3       15
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.2274 -0.3592  0.0027  0.6531  3.5841
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  location (Intercept) 2.682    1.638
## Number of obs: 20, groups:  location, 5
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.1799     0.7594   4.187 2.82e-05 ***
## period7      -2.1545     0.2368  -9.097  < 2e-16 ***
## period8      -2.3085     0.2414  -9.564  < 2e-16 ***
## period11     -2.4324     0.2325 -10.460  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) perid7 perid8
## period7  -0.224
## period8  -0.225  0.732
## period11 -0.234  0.758  0.761
```

```
emmeans(m3, ~period, type='response') %>% multcomp::cld(Letters=letters)
```

```
## period  prob     SE  df asymp.LCL asymp.UCL .group
## 11      0.679 0.1615 Inf     0.331     0.900  a
## 8       0.705 0.1546 Inf     0.358     0.911  a
## 7       0.736 0.1444 Inf     0.394     0.923  a
## 4       0.960 0.0291 Inf     0.844     0.991   b
##
## Confidence level used: 0.95
```

```
## Intervals are back-transformed from the logit scale
## P value adjustment: tukey method for comparing a family of 4 estimates
## Tests are performed on the log odds ratio scale
## significance level used: alpha = 0.05
```

```
# Interaction  model
m4 <- glmer(cbind(survive,perish) ~ period + (1 + period|location),
            family=binomial, data=troutegg)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.0184817 (tol = 0.002, component 1)
```

```
#summary(m4)
#emmeans(m4, ~period, type='response') %>% multcomp::cld(Letters=letters)
```

Unfortunately, we aren't able to fit the saturated model using random effects
because the numerical optimization function for finding MLE estimates failed
to converge.

11.7 ROC Curves

The dataset `faraway::wbca` comes from a study of breast cancer in Wisconsin.
There are 681 cases of potentially cancerous tumors of which 238 are actually
malignant (ie cancerous). Determining whether a tumor is really malignant is
traditionally determined by an invasive surgical procedure. The purpose of this
study was to determine whether a new procedure called 'fine needle aspiration',
which draws only a small sample of tissue, could be effective in determining
tumor status.

```
data('wbca', package='faraway')

# clean up the data
wbca <- wbca %>%
  mutate(Class = ifelse(Class==0, 'malignant', 'benign')) %>%
  dplyr::select(Class, BNucl, UShap, USize)

# Fit the model where Malignant is considered a success
# model <- glm( I(Class=='malignant') ~ ., data=wbca, family='binomial' )  # emmeans hates this

model <- wbca %>% mutate( Class = (Class == 'malignant') ) %>%   # Clear what is success
  glm( Class ~., data=., family='binomial' )                     # and emmeans still happy
```

```
# Get the response values
# type='response' gives phat values which live in [0,1]
# type='link' gives the Xbeta values whice live in (-infinity, infinity)
wbca <- wbca %>%
  mutate(phat = predict(model, type='response'),
         yhat = ifelse(phat > .5, 'malignant', 'benign'))

# Calculate the confusion matrix
table( Truth=wbca$Class, Predicted=wbca$yhat )
```

```
##           Predicted
## Truth       benign malignant
##   benign       432        11
##   malignant     15       223
```

As usual we can calculate the summary tables...

```
summary(model)
```

```
##
## Call:
## glm(formula = Class ~ ., family = "binomial", data = .)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8890  -0.1409  -0.1409   0.0287   2.2284
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.35433    0.54076 -11.751  < 2e-16 ***
## BNucl        0.55297    0.08041   6.877 6.13e-12 ***
## UShap        0.62583    0.17506   3.575 0.000350 ***
## USize        0.56793    0.15910   3.570 0.000358 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 881.39  on 680  degrees of freedom
## Residual deviance: 148.43  on 677  degrees of freedom
## AIC: 156.43
##
## Number of Fisher Scoring iterations: 7
```

From this table, we see that for a breast tumor, the larger values of `BNucl`, `UShap`, and `USize` imply a greater probability of it being malignant. So for a tumor with

```
##   BNucl UShap USize
## 1     2     1     2
```

We would calculate

$$\begin{aligned} X\hat{\beta} &= \hat{\beta}_0 + 2 \cdot \hat{\beta}_1 + 1 \cdot \hat{\beta}_2 + 2 \cdot \hat{\beta}_3 \\ &= -6.35 + 2*(0.553) + 1*(0.626) + 2*(0.568) \\ &= -3.482 \end{aligned}$$

and therefore

$$\hat{p} = \frac{1}{1 + e^{-X\hat{\beta}}} = \frac{1}{1 + e^{3.482}} = 0.0297$$

```
newdata = data.frame( BNucl=2, UShap=1, USize=2 )
predict(model, newdata=newdata)
```

```
##         1
## -3.486719
```

```
predict(model, newdata=newdata, type='response')
```

```
##         1
## 0.0296925
```

So for a tumor with these covariates, we would classify it as most likely to be benign.

In this medical scenario where we have to decide how to classify if a tumor is malignant of benign, we shouldn't treat the misclassification errors as being the same. If we incorrectly identify a tumor as malignant when it is not, that will cause a patient to undergo a somewhat invasive surgury to remove the tumor. However if we incorrectly identify a tumor as being benign, then the cancerous tumor will likely grow and eventually kill the patient. While the first error is regretable, the second is far worse.

Given that reasoning, perhaps we shouldn't use the rule: If $\hat{p} >= 0.5$ classify as malignant. Instead perhaps we should use $\hat{p} >= 0.3$ or $\hat{p} >= 0.05$.

Whatever decision rule we make, we should consider how many of each type of error we make. Consider the following confusion matrix:

	Predict Negative	Predict Positive	Total
Negative	True Neg (TN)	False Pos (FP)	N
Positive	False Neg (FN)	True Pos (TP)	P
Total	N^*	P^*	

where P is the number of positive cases, N is the number of negative cases, P^* is the number of observations predicted to be positive, and N^* is the number predicted to be negative.

Quantity	Formula	Synonyms
False Positive Rate	FP/N	Type I Error; 1-Specificity
True Positive Rate	TP/P	Power; Sensitivity; Recall
Pos. Pred. Value	TP/P^*	Precision

We can think of the True Positive Rate as the probability that a Positive case will be correctly classified as a positive. Similarly a False Positive Rate is the probability that a Negative case will be incorrectly classified as a positive.

I wish to examine the relationship between the False Positive Rate and the True Positive Rate for any decision rule. So what we could do is select a sequence of decision rules and for each calculate the `(FPR, TPR)` pair, and then make a plot where we play connect the dots with the `(FPR, TPR)` pairs.
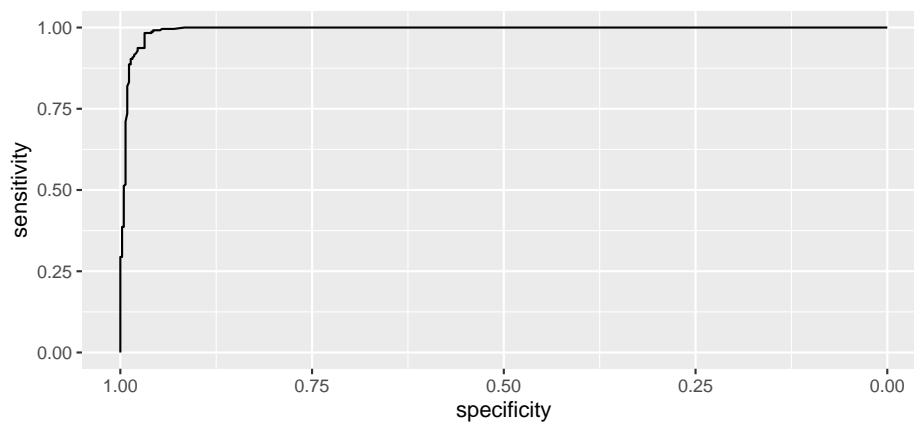
Of course we don't want to have to do this by hand, so we'll use the package `pROC` to do it for us.

```
# Calculate the ROC information using pROC::roc()
myROC <- roc( wbca$Class,  wbca$phat )
```

```
## Setting levels: control = benign, case = malignant
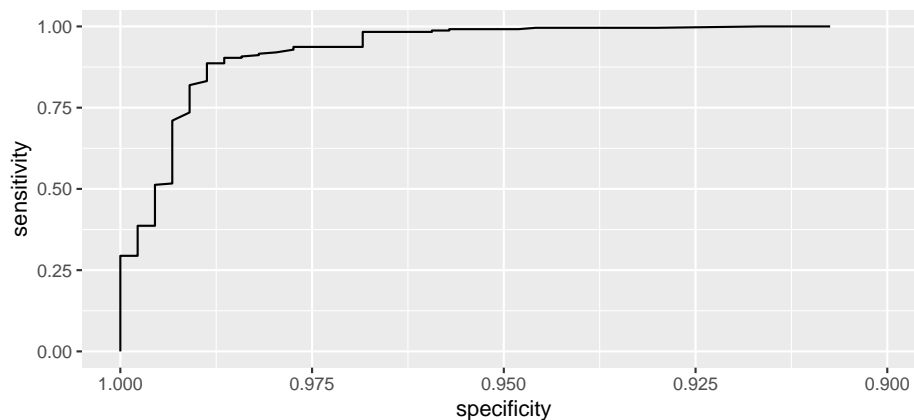```

```
## Setting direction: controls < cases
```

```
# make a nice plot using ggplot2 and pROC::ggroc()
ggroc( myROC )
```

This looks pretty good and in an ideal classifier that makes prefect predictions, this would be a perfect right angle at the bend.

Lets zoom in a little on the high specificity values (i.e. low false positive rates)

```
ggroc( myROC ) + xlim(1, .9)
```


We see that if we want to correctly identify about 99% of maligant tumors, we will have a false positive rate of about 1-0.95 = 0.05. So about 5% of benign tumors would be incorrectly classified as malignant.

It is a little challenging to read the graph to see what the Sensitivity is for a particular value of Specificity. To do this we'll use another function because the authors would prefer you to also estimate the confidence intervals for the quantity. This is a case where bootstrap confidence intervals are quite effective.

```
ci(myROC, of='sp', sensitivities=.99)
```

```
## 95% CI (2000 stratified bootstrap replicates):
```

```
##    se sp.low sp.median sp.high
##  0.99 0.9187    0.9571  0.9797
```

```
ci(myROC, of='se', specificities=.975)
```

```
## 95% CI (2000 stratified bootstrap replicates):
##     sp se.low se.median se.high
##  0.975 0.8827     0.937  0.9958
```

One measure of how far we are from the perfect predictor is the area under the curve. The perfect model would have an area under the curve of 1. For this model the area under the curve is:

```
auc(myROC)
```

```
## Area under the curve: 0.9929
```

```
ci(myROC, of='auc')
```

```
## 95% CI: 0.9878-0.9981 (DeLong)
```

```
ci(myROC, of='auc', method='bootstrap')
```

```
## 95% CI: 0.9872-0.9973 (2000 stratified bootstrap replicates)
```

which seems pretty good and Area Under the Curve (AUC) is often used as a way of comparing the quality of binary classifiers.

11.8   Exercises

1. The dataset `faraway::wbca` comes from a study of breast cancer in Wisconsin. There are 681 cases of potentially cancerous tumors of which 238 are actually malignant (i.e. cancerous). Determining whether a tumor is really malignant is traditionally determined by an invasive surgical procedure. The purpose of this study was to determine whether a new procedure called 'fine needle aspiration', which draws only a small sample of tissue, could be effective in determining tumor status.

a. Fit a binomial regression with `Class` as the response variable and the other nine variables as predictors (for consistency among students, define a success as the tumor being benign and remember that `glm` wants the response to be a matrix where the first column is the number of successes). Report the residual deviance and associated degrees of freedom. Can this information be used to determine if this model fits the data?

b. Use AIC as the criterion to determine the best subset of variables using the step function.

c. Use the reduced model to give the estimated probability that a tumor with associated predictor variables

```
newdata <- data.frame( Adhes=1, BNucl=1, Chrom=3, Epith=2, Mitos=1,
                       NNucl=1, Thick=4, UShap=1, USize=1)
```

is benign and give a confidence interval for your estimate.

d. Suppose that a cancer is classified as benign if $\hat{p} > 0.5$ and malignant if $\hat{p} \leq 0.5$. Compute the number of errors of both types that will be made if this method is applied to the current data with the reduced model. *Hint: save the $\hat{p}$ as a column in the wbca data frame and use that to create a new column* **Est_Class** *which is the estimated class (making sure it is the same encoding scheme as Class). Then use dplyr functions to create a table of how many rows fall into each of the four Class/Est_Class combinations.*

e. Suppose we changed the cutoff to 0.9. Compute the number of errors of each type in this case. Discuss the ethical issues in determining the cutoff.

2. Aflatoxin B1 was fed to lab animals at various doses and the number responding with liver cancer recorded and is available in the dataset `faraway::aflatoxin`.

a. Build a model to predict the occurrence of liver cancer. Consider a square-root transformation to the dose level.

b. Compute the ED50 level (effective dose level... same as LD50 but isn't confined to strictly lethal effects) and an approximate 95% confidence interval.

3. The dataset `faraway::pima` is data from a study of adult female Pima Indians living near Phoenix was done and resulted $n = 752$ observations after the cases of missing data (obnoxiously coded as 0) were removed. Testing positive for diabetes was the success (`test`) and the predictor variables we will use are: `pregnant`, `glucose`, and `bmi`.

a. Remove the observations that have missing data (coded as a zero) for either `glucose` or `bmi`. *The researcher's choice of using 0 to represent missing data is a bad idea because 0 is a valid value for the number*

*of pregnancies, so assume a zero in the **pregnant** covariate is a true value. The **dplyr** function **filter** could be used here.*

b. Fit the logistic regression model for `test` with using the main effects of `glucose`, `bmi`, and `pregnant`.

c. Produce a graphic that displays the relationship between the variables. *Notice I've done the part (a) for you and the assume that your model produced in part (b) is named **m**. I also split up the pregnancy and bmi values into some logical grouping for the visualization. If you've never used the **cut** function, go look it up because it is extremely handy.*

```r
pima <- pima %>% filter( bmi != 0, glucose != 0)
pima <- pima %>% mutate(
  phat=ilogit(predict(m)),
  pregnant.grp = cut(pregnant, c(0,1,3,6,100), right = FALSE, labels = c('0',
  bmi.grp = cut(bmi, c(0,18,25,30,100), labels=c('Underweight','Normal','Overu
ggplot(pima, aes(y=test, x=glucose)) +
  geom_point() +
  geom_line(aes(y=phat), color='red') +
  facet_grid(bmi.grp ~ pregnant.grp)
```

d. Discuss the quality of your predictions based on the graphic above and modify your model accordingly.

e. Give the probability of testing positive for diabetes for a Pima woman who had had no pregnancies, had `bmi=28` and a glucose level of `110`.

f. Give the odds that the same woman would test positive for diabetes.

g. How do her odds change to if she were to have a child? That is to say, what is the odds ratio for that change?

Appendix

Chapter 12

Block Designs

```r
# packages for this chapter
library(tidyverse)    # ggplot2, dplyr, etc...
library(emmeans)      # TukeyLetters stuff
```

Often there are covariates in the experimental units that are known to affect the response variable and must be taken into account. Ideally an experimenter can group the experimental units into blocks where the within block variance is small, but the block to block variability is large. For example, in testing a drug to prevent heart disease, we know that gender, age, and exercise levels play a large role. We should partition our study participants into gender, age, and exercise groups and then randomly assign the treatment (placebo vs drug) within the group. This will ensure that we do not have a gender, age, and exercise group that has all placebo observations.

Often blocking variables are not the variables that we are primarily interested in, but must nevertheless be considered. We call these nuisance variables. We already know how to deal with these variables by adding them to the model, but there are experimental designs where we must be careful because the experimental treatments are *nested*.

Example 1. An agricultural field study has three fields in which the researchers will evaluate the quality of three different varieties of barley. Due to how they harvest the barley, we can only create a maximum of three plots in each field. In this example we will block on field since there might be differences in soil type, drainage, etc from field to field. In each field, we will plant all three varieties so that we can tell the difference between varieties without the block effect of field confounding our inference. In this example, the varieties are nested within the fields.
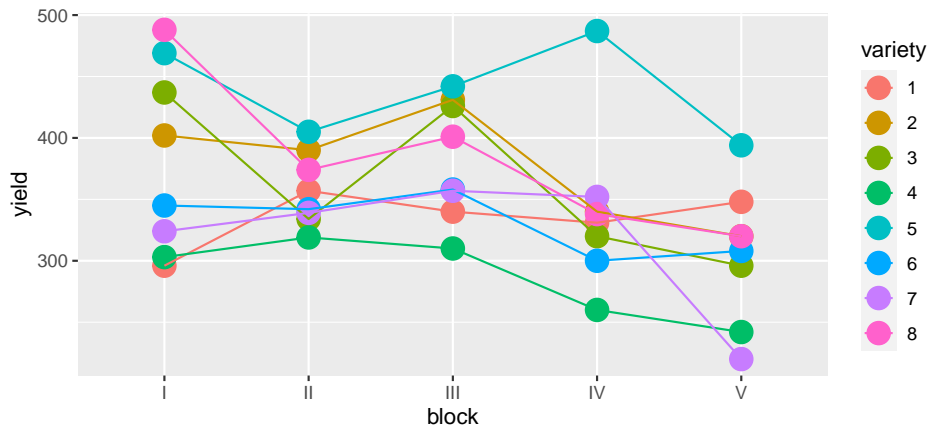
257

	Field 1	Field 2	Field 3
Plot 1	Variety A	Variety C	Variety B
Plot 2	Variety B	Variety A	Variety C
Plot 3	Variety C	Variety B	Variety A

Example 2. We are interested in how a mouse responds to five different materials inserted into subcutaneous tissue to evaluate the materials' use in medicine. Each mouse can have a maximum of 3 insertions. Here we will block on the individual mice because even lab mice have individual variation. We actually are not interested in estimating the effect of the mice because they aren't really of interest, but the mouse block effect should be accounted for before we make any inferences about the materials. Notice that if we only have one insertion per mouse, then the mouse effect will be confounded with materials.

12.1 Randomized   Complete   Block   Design (RCBD)

The dataset `oatvar` in the faraway library contains information about an experiment on eight different varieties of oats. The area in which the experiment was done had some systematic variability and the researchers divided the area up into five different blocks in which they felt the area inside a block was uniform while acknowledging that some blocks are likely superior to others for growing crops. Within each block, the researchers created eight plots and randomly assigned a variety to a plot. This type of design is called a Randomized Complete Block Design (RCBD) because each block contains all possible levels of the factor of primary interest.

```
data('oatvar', package='faraway')
ggplot(oatvar, aes(y=yield, x=block, color=variety)) +
    geom_point(size=5) +
    geom_line(aes(x=as.integer(block))) # connect the dots
```

While there is one unusual observation in block IV, there doesn't appear to be a blatant interaction. We will consider the interaction shortly. For the main effects model of yield ~ block + variety we have $p = 12$ parameters and 28 residual degrees of freedom because

$$df_\epsilon = n - p$$
$$= n - (1 + [(I - 1) + (J - 1)])$$
$$= 40 - (1 + [(5 - 1) + (8 - 1)])$$
$$= 40 - 12$$
$$= 28$$

```
m1 <- lm( yield ~ block + variety, data=oatvar)
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: yield
##            Df Sum Sq Mean Sq F value    Pr(>F)
## block       4  33396  8348.9  6.2449  0.001008 **
## variety     7  77524 11074.8  8.2839 1.804e-05 ***
## Residuals  28  37433  1336.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# plot(m1)       # check diagnostic plots - they are fine...
```

Because this is an orthogonal design, the sums of squares doesn't change regardless of which order we add the factors, but if we remove one or two observations, they would.

In determining the significance of `variety` the above F-value and p-value is correct. We have 40 observations (5 per variety), and after accounting for the
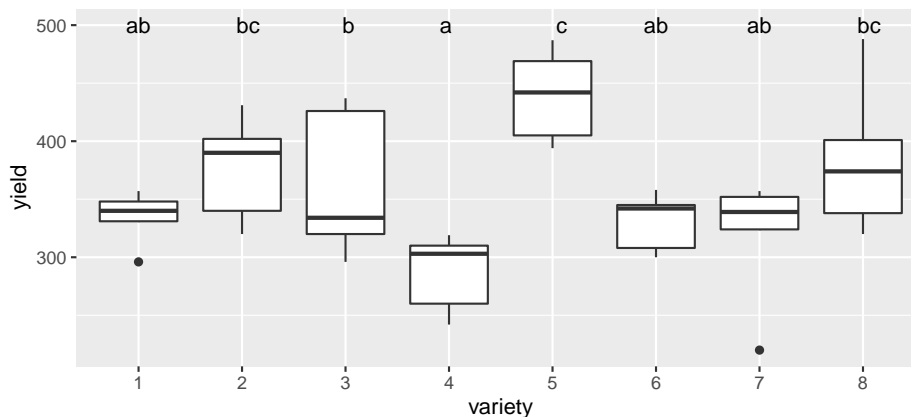
model structure (including the extraneous blocking variable), we have 28 residual degrees of freedom.

But the F-value and p-value for testing if `block` is significant is nonsense! Imagine that variety didn't matter we just have 8 replicate samples per block, but these aren't true replicates, they are what is called *pseudoreplicates*. Imagine taking a sample of $n = 3$ people and observing their height at 1000 different points in time during the day. You don't have 3000 data points for estimating the mean height in the population, you have 3. Unless we account for the this, the inference for the block variable is wrong. In this case, we only have one observation for each block, so we can't do any statistical inference at the block scale!

Fortunately in this case, we don't care about the blocking variable and including it in the model was simply guarding us in case there was a difference, but I wasn't interested in estimating it. If the only covariate we care about is the most deeply nested effect, then we can do the usual analysis and recognize the p-value for the blocking variable is nonsense, and we don't care about it.

```
# Ignore any p-values regarding block, but I'm happy with the analysis for variety
letter_df <- emmeans(m1, ~variety) %>%
  multcomp::cld(Letters=letters) %>%
  dplyr::select(variety, .group) %>%
  mutate(yield = 500)

ggplot(oatvar, aes(x=variety, y=yield)) +
  geom_boxplot() +
  geom_text( data=letter_df, aes(label=.group) )
```


However it would be pretty sloppy to not do the analysis correctly because our blocking variable might be something we care about. To make R do the correct analysis, we have to denote the nesting. In this case we have block-to-block errors, and then variability within blocks. To denote the nesting we

use the `Error()` function within our formula. By default, `Error()` just creates independent error terms, but when we add a covariate, it adds the appropriate nesting.

```
m3 <- aov( yield ~ variety + Error(block), data=oatvar)
summary(m3)
```

```
##
## Error: block
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  4  33396    8349
##
## Error: Within
##           Df Sum Sq Mean Sq F value  Pr(>F)
## variety    7  77524   11075   8.284 1.8e-05 ***
## Residuals 28  37433    1337
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice that in our block level, there is no p-value to assess if the blocks are different. This is because we don't have any replication of the blocks. So our analysis respects that blocks are present, but does not attempt any statistical analyses on them.

12.2 Split-plot designs

There are plenty of experimental designs where we have levels of treatments nested within each other for practical reasons. The literature often gives the example of an agriculture experiment where we investigate the effect of irrigation and fertilizer on the yield of a crop. However because our irrigation system can't be fine-tuned, we have plots with different irrigation levels and within each plot we have perhaps four subplots that have the fertilizer treatment. To summarize, Irrigation treatments were randomly assigned to plots, and fertilizer treatments were randomly assigned to sub-plots.

```
## `summarise()` regrouping output by 'plot', 'subplot', 'Fertilizer' (override with `.groups` ar
```

```
## # A tibble: 6 x 5
## # Groups:   plot, subplot, Fertilizer [6]
##   plot  subplot Fertilizer Irrigation yield
##   <fct> <fct>   <fct>      <fct>      <dbl>
## 1 1     1       Low        Low         20.2
## 2 1     2       High       Low         24.4
```

```
## 3 1      3         Low          Low            18.0
## 4 1      4         High         Low            21.0
## 5 2      1         Low          Low            23.2
## 6 2      2         High         Low            26.5
```


So all together we have 8 plots, and 32 subplots. When I analyze the fertilizer, I have 32 experimental units (the thing I have applied my treatment to), but when analyzing the effect of irrigation, I only have 8 experimental units.

I like to think of this set up as having some lurking variables that act at the plot level (changes in aspect, maybe something related to what was planted prior) and some lurking variables that act on a local subplot scale (maybe variation in clay/silt/sand ratios). So even after I account for Irrigation and Fertilizer treatments, observations within a plot will be more similar to each other than observations in two different plots.

We can think about doing two separate analyses, one for the effect of irrigation, and another for the effect of the fertilizer.

```r
# AgData came from my data package, dsData, (however I did some summarization
# first.)

# To analyze Irrigation, average over the subplots first...
Irrigation.data <-   AgData %>%
  group_by(plot, Irrigation) %>%
  summarise( yield = mean(yield)) %>%
  as.data.frame()  # the aov command doesn't like tibbles.
```

```
## `summarise()` regrouping output by 'plot' (override with `.groups` argument)
```

```
# Now do a standard analysis. I use the aov() command instead of lm()
# because we will shortly do something very tricky that can only be
# done with aov(). For the most part, everything is
# identical from what you are used to.
m <- aov( yield ~ Irrigation, data=Irrigation.data )
anova(m)
```

```
## Analysis of Variance Table
##
## Response: yield
##            Df Sum Sq Mean Sq F value Pr(>F)
## Irrigation  1 26.064 26.0645  3.4281 0.1136
## Residuals   6 45.619  7.6032
```

In this case we see that we have insufficient evidence to conclude that the observed difference between the Irrigation levels could not be due to random chance.

Next we can do the appropriate analysis for the fertilizer, recognizing that all the p-values for the plot effects are nonsense and should be ignored.

```
m <- aov( yield ~ plot + Fertilizer, data=AgData )
summary(m)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## plot         7 286.73   40.96   3.153 0.0173 *
## Fertilizer   1   0.43    0.43   0.033 0.8572
## Residuals   23 298.83   12.99
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ideally I wouldn't have to do the averaging over the nested observations and we would like to not have the misleading p-values for the plots. To do this, we only have to specify the nesting of the error terms and R will figure out the appropriate degrees of freedom for the covariates.

```
# To do this right, we have to abandon the general lm() command and use the more
# specialized aov() command.  The Error() part of the formula allows me to nest
# the error terms and allow us to do the correct analysis. The order of these is
# to start with the largest/highest level and then work down the nesting.
m2 <- aov( yield ~ Irrigation + Fertilizer + Error(plot/subplot), data=AgData )
summary(m2)
```

```
##
## Error: plot
##           Df Sum Sq Mean Sq F value Pr(>F)
## Irrigation  1  104.3  104.26   3.428  0.114
## Residuals   6  182.5   30.41
##
## Error: plot:subplot
##           Df Sum Sq Mean Sq F value Pr(>F)
## Fertilizer  1   0.43    0.43   0.033  0.857
## Residuals  23 298.83   12.99
```

In the output, we see that the ANOVA table row for the Fertilizer is the same for both analyses, but the sums-of-squares for Irrigation are different between the two analyses (because of the averaging) while the F and p values are the same between the two analyses.

What would have happened if we had performed the analysis incorrectly and had too many degrees of freedom for the Irrigation test?

```r
bad.model <- aov( yield ~ Irrigation + Fertilizer, data=AgData)
anova(bad.model)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value  Pr(>F)
## Irrigation  1 104.26 104.258  6.2818 0.01806 *
## Fertilizer  1   0.43   0.430  0.0259 0.87324
## Residuals  29 481.31  16.597
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this case we would have concluded that we had statistically significant evidence to conclude the Irrigation levels are different. Notice that the sums-of-squares in this **wrong** analysis match up with the sums-of-squares in the correct design and the only difference is that when we figure out the sum-of-squares for the residuals we split that into different pools.

$$RSS_{total} = RSS_{Fertilizer} + RSS_{Irrigation}$$
$$456.12 = 273.64 + 182.5$$

When we want to infer if the amount of noise explained by adding Irrigation or Fertilizer is sufficiently large to justify their inclusion into the model, we compare the sum-of-squares value to the RSS but now we have to use the appropriate pool.

A second example of a slightly more complex split plot is given in the package `MASS` under the dataset `oats`. From the help file the data describes the following experiment:

> The yield of oats from a split-plot field trial using three varieties and four levels of manurial treatment. The experiment was laid out in 6 blocks of 3 main plots, each split into 4 sub-plots. The varieties were applied to the main plots and the manurial treatments to the sub-plots.

This is a lot to digest so lets unpack it. First we have 6 blocks and we'll replicate the exact same experiment in each block. Within a block, we'll split it into three sections, which we'll call plots (within the block). Finally within each plot, we'll have 4 subplots.

We have 3 varieties of oats, and 4 levels of fertilizer (manure). To each set of 3 plots, we'll randomly assign the 3 varieties, and to each set of subplots, we'll assign the fertilizers.

One issue that makes this issue confusing for students is that most texts get lazy and don't define the blocks, plots, and sub-plots when there are no replicates in a particular level. I prefer to be clear about defining those so.

```r
data('oats', package='MASS')
oats <- oats %>% mutate(
  Nf = ordered(N, levels = sort(levels(N))),   # make manure an ordered factor
  plot = as.integer(V),                        # plot
  subplot = as.integer(Nf))                    # sub-plot
```

As always we first create a graph to examine the data

```r
oats <- oats %>% mutate(B_Plot = interaction(B, plot))
ggplot(oats, aes(x=Nf, y=Y, color=V)) +
  facet_grid( B ~ plot, labeller=label_both) +
  geom_point() +
  geom_line(aes(x=as.integer(Nf)))
```
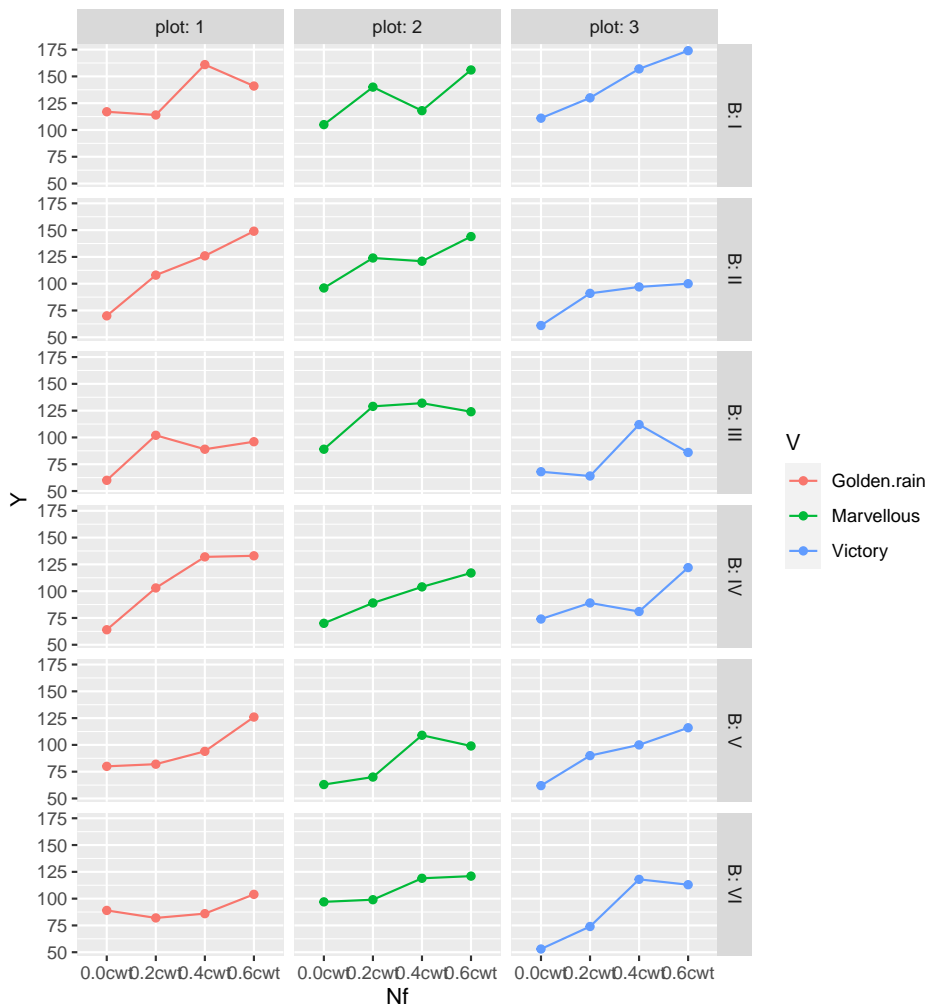
This graph also makes me think that variety doesn't matter and it is unlikely that there an interaction between oat variety and fertilizer level, but we should check.

```
#  What makes sense to me
# m.c <- aov( Y ~ V * Nf + Error(B/plot/subplot), data=oats)
```

Unfortunately the above model isn't correct because R isn't smart enough to understand that the levels of plot and subplot are exact matches to the Variety and Fertilizer levels. As a result if I defined the model above, the degrees of freedom will be all wrong because there is too much nesting. So we have to be smart enough to recognize that plot and subplot are actually Variety and Fertilizer.

```
m.c <- aov( Y ~ V * Nf + Error(B/V/Nf), data=oats)
summary(m.c)
```

```
##
## Error: B
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  5  15875    3175
##
## Error: B:V
##           Df Sum Sq Mean Sq F value Pr(>F)
## V          2   1786   893.2   1.485  0.272
## Residuals 10   6013   601.3
##
## Error: B:V:Nf
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Nf         3  20020    6673  37.686  2.46e-12 ***
## V:Nf       6    322      54   0.303     0.932
## Residuals 45   7969     177
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Sure enough the interaction term is not significant. We next consider the Variety term.

```
m.s <- aov( Y ~ V + Nf + Error(B/V/Nf), data=oats)
summary(m.s)
```

```
##
## Error: B
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  5  15875    3175
##
## Error: B:V
##           Df Sum Sq Mean Sq F value Pr(>F)
## V          2   1786   893.2   1.485  0.272
## Residuals 10   6013   601.3
##
## Error: B:V:Nf
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Nf         3  20020    6673   41.05  1.23e-13 ***
## Residuals 51   8291     163
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We conclude by noticing that the Variety does not matter, but that the fertilizer level is quite significant.

––––––––––––––––––––––––

There are many other types of designs out there. For example you might have 5 levels of a factor, but when you split your block into plots, you can only create 3 plots. So not every block will have every level of the factor. This is called *Randomized Incomplete Block Designs* (RIBD).

You might have a design where you apply even more levels of nesting. Suppose you have a green house study where you have rooms where you can apply a temperature treatment, within the room you have four tables and can apply a light treatment to each table. Finally within each table you can have four trays where can apply a soil treatment to each tray. This is a continuation of the split-plot design and by extending the nesting we can develop *split-split-plot* and *split-split-split-plot* designs.

You might have 7 covariates each with two levels (High, Low) and you want to investigate how these influence your response but also allow for second and third order interactions. If you looked at every treatment combination you'd have $2^7 = 128$ different treatment combinations and perhaps you only have the budget for a sample of $n = 32$. How should you design your experiment? This question is addressed by *fractional factorial* designs.

If your research interests involve designing experiments such as these, you should consider taking an Experimental design course.

12.3   Exercises

1. ???
2. ???
3. ???

Chapter 13

Maximum Likelihood Estimation

```r
library(tidyverse)   # dplyr, tidyr, ggplot2
```

Learning Outcomes

- Explain how the probability mass/density function $f(x|\theta)$ indicates what data regions are more probable.
- Explain how the likelihood function $\mathcal{L}(\theta|x)$ is defined if we know the probability function.
- Explain how the likelihood function $\mathcal{L}(\theta|x)$ is used to find the maximum likelihood estimate of θ.
- For a given sample of data drawn from a distribution, find the maximum likelihood estimate for the distribution parameters using R.

13.1 Introduction

The goal of statistical modeling is to take data that has some general trend along with some un-explainable variability, and say something intelligent about the trend. For example, the simple regression model

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \qquad \text{where} \;\; \epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$$
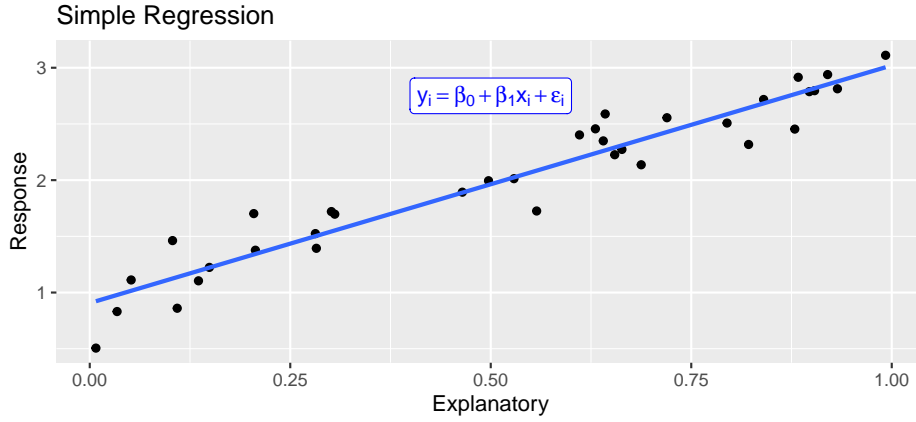
Simple Regression

There is a general increasing trend in the response (i.e. the $\beta_0 + \beta_1 x_i$ term) and then some un-explainable noise (the $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ part).

While it has been convenient to write the model in this form, it is also possible to write the simple regression model as

$$y_i \overset{ind}{\sim} N(\,\beta_0 + \beta_1 x_i, \; \sigma^2\,)$$

This model contains three parameters β_0, β_1, and σ but it certainly isn't clear how to estimate these three values. In this chapter, we'll develop a mechanism for taking observed data sampled from some distribution parameterized by some β, λ, σ, or θ and then estimating those parameters.

13.2   Distributions

Depending on what values the data can take on (integers, positive values) and the shape of the distribution of values, we might chose to model the data using one of several different distributions. Next we'll quickly introduce the mathematical relationship between the parameter and probable data values of several distributions.

13.2.1   Poisson

The Poisson distribution is used to model the number of events that happen in some unit of time or space. This distribution is often used to model events that can only be positive integers. This distribution is parameterized by λ, which represents the expected number of events that happen (as defined as the average over an infinitely large number of draws). Because λ represents the average number of events, the λ parameter must be greater than or equal to 0.

The function that defines the relationship between the parameter λ and what values are most probable is called the *probability mass function* when talking about discrete random variables and *probability density functions* in the continuous case. Either way, these functions are traditionally notated using $f(x)$.

$$f(x|\lambda) = \frac{e^{-\lambda}\lambda^x}{x!} \quad \text{for } x \in \{0, 1, 2, ...\}$$

Poisson ($\lambda = 3.5$)
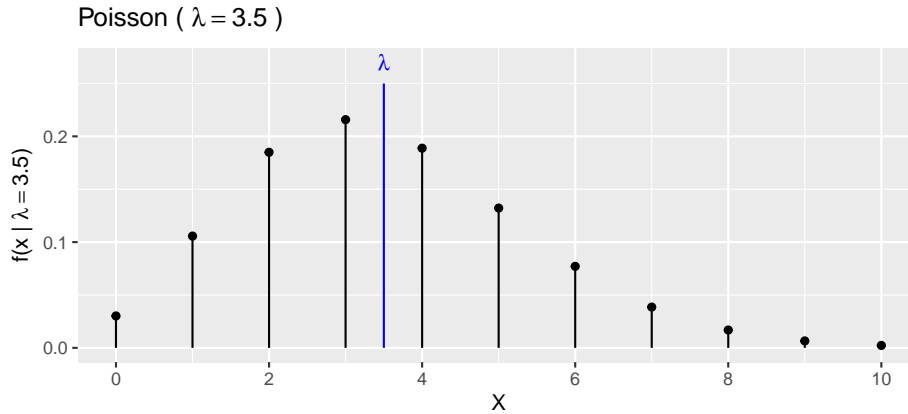
The notation $f(x|\lambda)$ read as "f *given* λ" and is used to denote that this is a function that describes what values of the data X are most probable and that the function depends on the parameter value. This is emphasizing that if we were to change the parameter value (to say $\lambda = 10$), then a different set of data values would be more probable. In the above example with $\lambda = 3.5$, the most probable outcome is 3 but we aren't surprised if we were to observe a value of $x = 1, 2$, or 4. However, from this graph, we see that $x = 10$ or $x = 15$ would be highly improbable.

13.2.2   Exponential

The Exponential distribution can be used to model events that take on a positive real value and the distribution of values has some skewness. We will parameterize this distribution using β as the mean of the distribution.

$$f(x|\beta) = \beta e^{\beta x} \quad \text{for } x > 0$$

Exponential (β = 3.5)
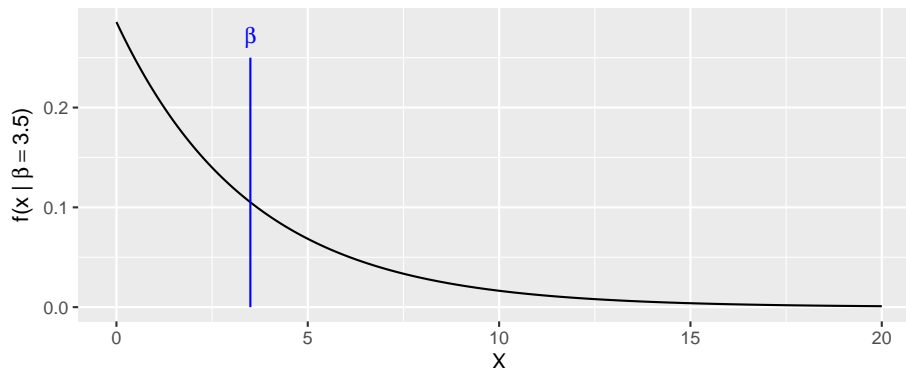
In this distribution, the region near zero is the most probable outcome, and larger observations are less probable.

13.2.3   Normal

The Normal distribution is extremely commonly used in statistics and can be used to modeled continuous variables and is parameterized by the center μ and spread σ parameters.

$$f(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

where $\exp[w] = e^w$ is just a notational convenience.

Normal (μ = 6, σ = 1)
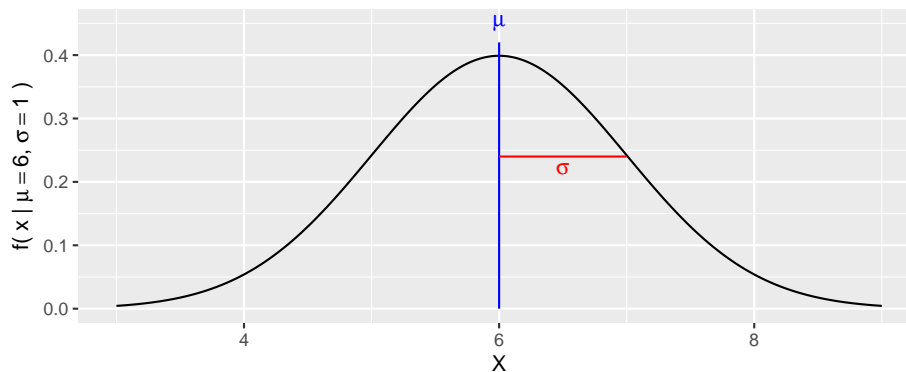
All of these distributions (and there are many, many more distributions commonly used) have some mathematical function that defines the how probable a region of response values is and that function depends on the parameters. Importantly X regions with the highest $f(x|\theta)$ are the most probable data values.

There are many additional mathematical details that go into these density functions but the important aspect is that they tell us what data values are most probable given some parameter values θ.

13.3 Likelihood Function

As a researcher, I am not particularly interested in saying "If $\mu = 3$ and $\sigma = 2$ then I'm likely to observe approximately 95% of my data between -1 and 7. Instead, I want to make an inference about what values for μ and σ are the most concordant with observed data that I've collected. However, the probability density function $f(x|\mu, \sigma)$ is still the mathematical link between the data and parameter and we will continue to use that function, but we'll re-interpret which is *known*.

The *Likelihood* function is just the probability density (or mass) function $f(x|\theta)$ re-interpreted to be a function where the data is the known quantity and we are looking to see what parameter values are consistent with the data.

13.3.1 Poisson

Suppose that we have observed a single data point drawn from a Poisson(λ) and we don't know what λ is. We first write down the likelihood function

$$\mathcal{L}(\lambda|x) = f(x|\lambda) = \frac{e^{-\lambda}\lambda^x}{x!}$$

If we have observed $x = 4$, then $\mathcal{L}(\lambda|x = 4)$ is the following function

Poisson Likelihood given x=4

Poisson log–Likelihood given x=4
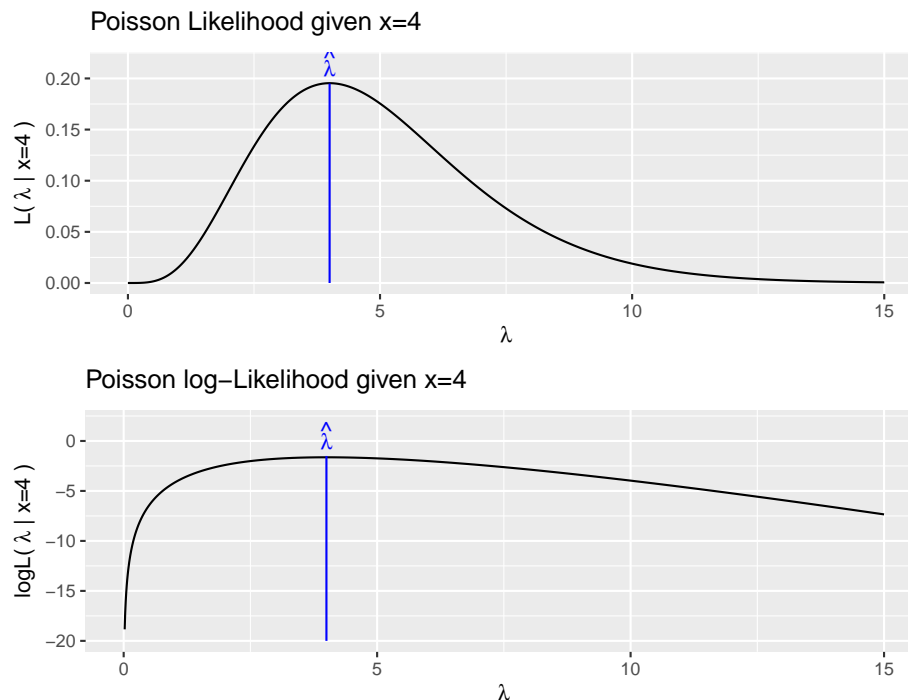
Our best estimate for λ is the value that maximizes $\mathcal{L}(\lambda|x)$. We could do this two different ways. First we could mathematically solve by taking the derivative, setting it equal to zero, and then solving for lambda. Often this process is made mathematically simpler (and computationally more stable) by instead maximizing the log of the Likelihood function. This is equivalent because the log function is monotonically increasing and if $a < b$ then $\log(a) < \log(b)$. It is simpler because taking logs makes everything 1 operation simpler and reduces the need for using the chain rule while taking derivatives. We could also find the value of λ that maximizes the likelihood using numerical methods. Again because the log function makes everything nicer, in practice we'll always maximize the log likelihood. Many optimization functions are designed around finding function minimums, so to use those, we'll actually seek to minimize the *negative* log likelihood which is simply $-1 * \log \mathcal{L}()$.

Numerical solvers are convenient, but are only accurate to machine tolerance you specify. In this case where $x = 4$, the actual maximum likelihood value is $\hat{\lambda} = 4$.

```r
x <- 4
neglogL <- function(param){
  dpois(x, lambda=param) %>%
    log() %>%      # take the log
    prod(-1) %>%  # multiply by -1
    return()
```

```
}
```

```
# Optimize function will find the maximum of the Likelihood function
# over the range of lambda values [0, 20]. By default, the optimize function
# finds the minimum, but has an option to find the maximum. Alternatively
# we could find the minimum of the -logL function.
optimize(neglogL, interval=c(0,20) )
```

```
## $minimum
## [1] 3.999993
##
## $objective
## [1] 1.632876
```

But what if we have multiple observations from this Poisson distribution? If the observations are *independent*, then the probability mass or probability density functions $f(x_i|\theta)$ can just be multiplied together.

$$\mathcal{L}(\lambda|\mathbf{x}) = \prod_{i=1}^{n} f(x_i|\lambda) = \prod_{i=1}^{n} \frac{e^{-\lambda}\lambda_i^x}{x_i!}$$

So, suppose we have observed $\mathbf{x} = \{4, 6, 3, 3, 2, 4, 3, 2\}$. We could maximize this function either using calculus methods or numerical methods this function and discover that the maximum occurs at $\hat{\lambda} = \bar{x} = 3.375$.

Poisson Likelihood given **x**

Poisson log–Likelihood given **x**
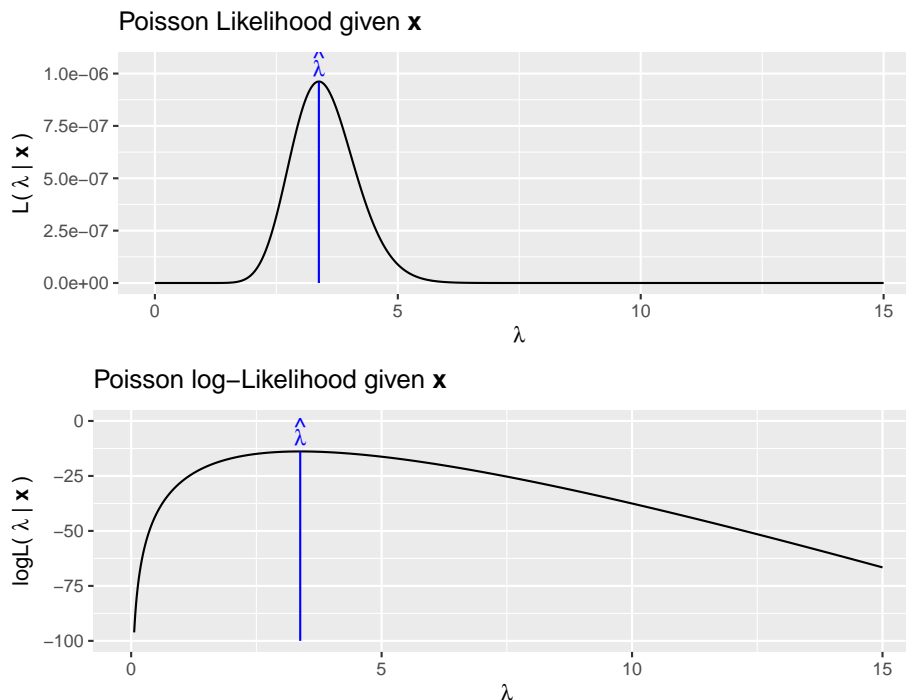
If we are using the log-Likelihood, then the multiplication is equivalent to summing and we'll define and subsequently optimize our log-Likelihood function like this:

```r
x <- c(4,6,3,3,2,4,3,2)
neglogL <- function(param){
  dpois(x, lambda=param) %>%
    log() %>% sum() %>% prod(-1) %>%
    return()
}
optimize(neglogL, interval=c(0,20) )
```

```
## $minimum
## [1] 3.37499
##
## $objective
## [1] 13.85426
```

13.3.2   Exponential Example

We next consider data sampled from the exponential distribution. Recall the exponential distribution can be parametrized by a single parameter, β and

which is the expectation of the distribution and variance is β^2. We might consider using as an estimator *either* the sample mean or the sample standard deviation. It turns out that the sample mean is the maximum likelihood estimator in this case. For a concrete example, suppose that we had observed $\mathbf{x} = \{15.6, 2.03, 9.12, 1.54, 3.69\}$

The likelihood is

$$\mathcal{L}(\beta|\mathbf{x}) = \prod_{i=1}^{n} \beta e^{\beta x_i}$$

A bit of calculus will show that this is maximized at $\hat{\beta} = \bar{x} = 6.39$. We can numerically see this following the same process as previously seen.

```
## [1] 6.396
```


Exponential Likelihood given **x**

Exponential log–Likelihood given **x**
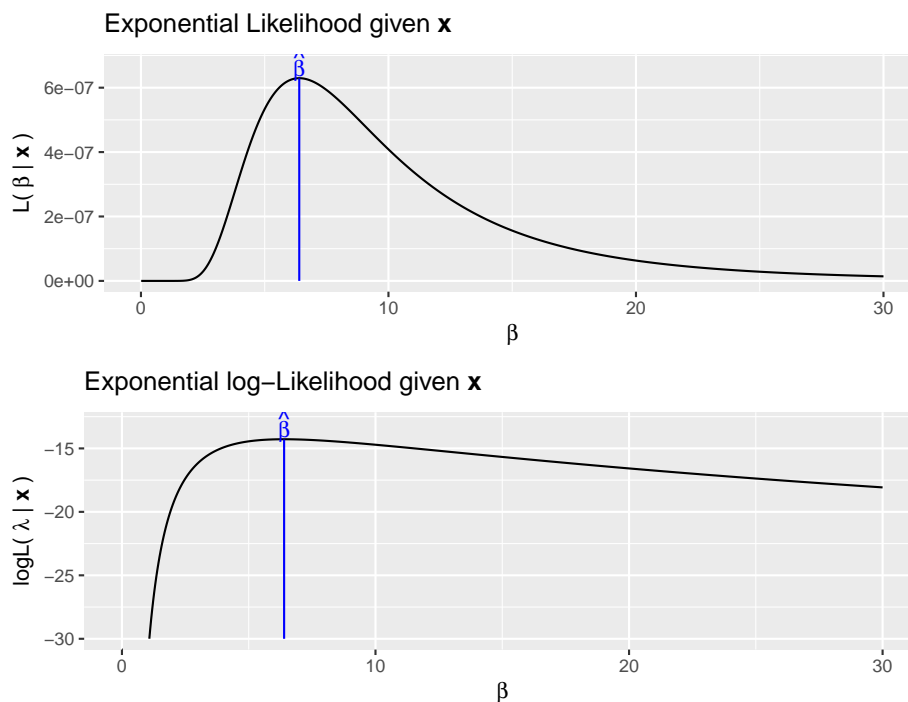
```
x <- c(15.6, 2.03, 9.12, 1.54, 3.69)
neglogL <- function(param){
  dexp(x, rate=1/param, log = TRUE) %>%    # We defined beta as 1/rate
    sum() %>%
    prod(-1) %>%
    return()
}
optimize(neglogL, interval=c(0,30) )
```

```
## $minimum
## [1] 6.396004
##
## $objective
## [1] 14.27836
```

13.3.3   Normal

We finally consider the case where we have observation coming from a distribution that has multiple parameters. The normal distribution is parameterized by a mean μ and spread σ. Suppose that we had observed $x_i \overset{iid}{\sim} N(\mu, \sigma^2)$ and saw $\mathbf{x} = \{5, 8, 9, 7, 11, 9\}$.

As usual we can calculate the likelihood function

$$\mathcal{L}(\mu, \sigma | \mathbf{x}) = \prod_{i=1}^{n} f(x_i | \mu, \sigma) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\frac{(x_i - \mu)^2}{\sigma^2}\right]$$
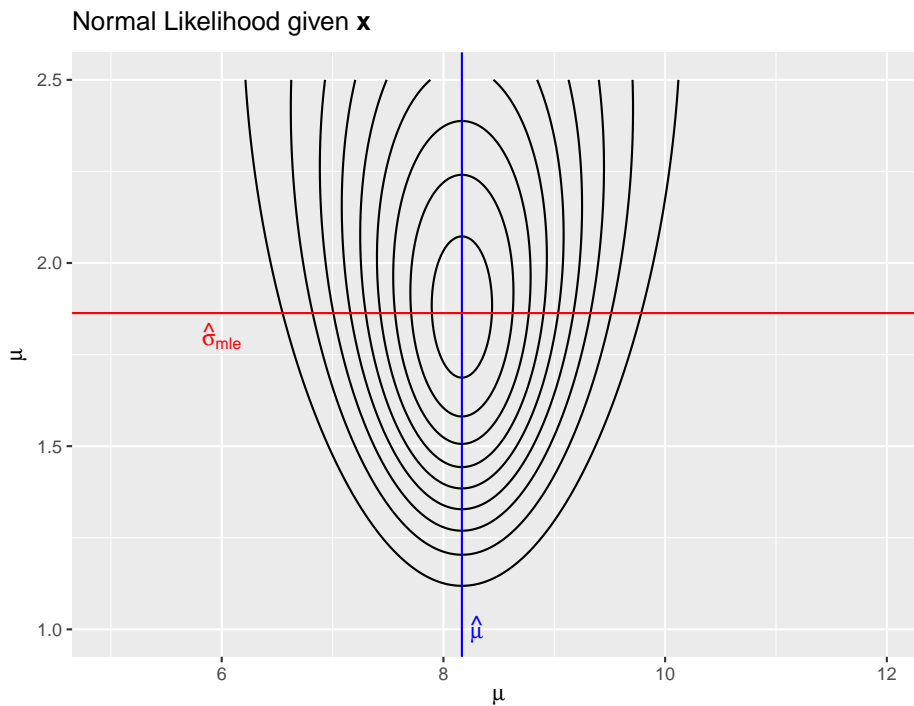
Again using calculus, it can be shown that the maximum likelihood estimators in this model are

$$\hat{\mu} = \bar{x} = 8.16666$$

$$\hat{\sigma}_{mle} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} = 1.8634$$

which is somewhat unexpected because the typical estimator we use has a $\frac{1}{n-1}$ multiplier.

```
x <- c(5, 8, 9, 7, 11, 9)
xbar <- mean(x)
s2 <- sum( (x - xbar)^2 / 6 )
s <- sqrt(s2)
```

Normal Likelihood given **x**


```
x <- c(5, 8, 9, 7, 11, 9)
neglogL <- function(param){
  dnorm(x, mean=param[1], sd=param[2]) %>%
    log() %>% sum() %>% prod(-1) %>%
    return()
}

# Bivariate optimization uses the optim function that only can search
# for a minimum. The first argument is an initial guess to start the algorithm.
# So long as the start point isn't totally insane, the numerical algorithm should
# be fine.
optim(c(5,2), neglogL )
```

```
## $par
## [1] 8.166924 1.863391
##
## $value
## [1] 12.24802
##
## $counts
## function gradient
##       71       NA
```

```
##
## $convergence
## [1] 0
##
## $message
## NULL
```

13.4   Discussion

1. How could the numerical maximization happen? Assume we have a 1-dimensional parameter space, we have a reasonable starting estimate, and the function to be maximized is continuous, smooth, and ≥ 0 for all x. *While knowing the derivative function $f'(x)$ would allow us to be much more clever, lets think about how to do the maximization by just evaluating $f(x)$ for different values of x.*

2. Convince yourself that if x_0 is the value of x that maximizes $f(x)$, then it is also the value that maximizes $log(f(x))$. This will rest on the idea that $log()$ is a strictly increasing function.

13.5   Exercises

1. The χ^2 distribution is parameterized by its degrees of freedom parameter ν which corresponds to the mean of the distribution (ν must be > 0). The density function $f(x|\nu)$ can be accessed R using the `dchisq(x, df=nu)`.

   a) For different values of ν, plot the distribution function of $f(x|\nu)$. You might consider $\nu = 5, 10$, and 20. The valid range of x values is $[0, \infty)$ so select the

   b) Suppose that we've observed $x = \{9, 7, 7, 6, 10, 7, 9\}$. Calculate the sample mean and standard deviation.

   c) Graphically show that the maximum likelihood estimator of ν is $\bar{x} = 7.857$.

   d) Show that the maximum likelihood estimator of ν is $\bar{x} = 7.857$ using a numerical maximization function.

2. The Beta distribution is often used when dealing with data that are proportions. It is parameterized by *two* parameters, usually called α and β (both of which must be greater than zero). The mean of this distribution is by
$$E[X] = \frac{\alpha}{\alpha + \beta}$$
while the spread of the distribution is inversely related to the magnitude of α and β. The density function $f(x|\alpha, \beta)$ can be accessed in R using the `dbeta(x, alpha, beta)` function.

a) For different values of α and β, plot the distribution function $f(x|\alpha, \beta)$. You might consider keeping the ratio between α and β constant and just increase their magnitude. The valid range of x values is $[0, 1]$.

b) Suppose that we've observed $x = \{0.25, 0.42, 0.45, 0.50, 0.55\}$. Calculate the sample mean and standard deviation.

c) Graphically show that the maximum likelihood estimators of α and β are approximately 9.5 and 12.5.

d) Show that the maximum likelihood estimators of α and β are approximately 9.5 and 12.5.

Project Appendix

STA 571 Course Project Timeline This document presents a rough outline to the projects conducted in STA 571: Statistical Methods II. The outline gives when certain topics should be discussed, when decisions should be made on moving forward with projects, and what students need to be focused on. Proposal and final project rubrics will be provided in additional PDFs. As always, there is flexibility based on the way you run your course.

13.6 Weeks 1 − 4 (Project Feasibility)

Leading up to Exam 1, decisions will be made if the projects can be conducted effectively within the course. This requires students to determine what data set they want to analyze and what methods they think they would be interested in using. 1. For students with thesis/dissertation work, the idea is to allow them to use their own data (likely data they have collected) to advance the project. 2. Students that don't have thesis/dissertation work, or for students who don't yet have data from their own research should be engaged in looking into a source of data and determining what type of analysis can be completed. One very good source of datasets and problems is kaggle.

13.6.1 WIBGIs

Project ideas can be motivated through "Wouldn't it be great if …" statements, or WIBGIs.

Write out 3 - 5 ideas for a project to be done this semester. Think about what data you may have available or what data you may be interested in analyzing. If possible try to write a statement of this form:

Wouldn't it be great if what you want to analyze could be used to solve a problem using statistics/data science.

Example: Wouldn't it be great if plasma thermogram data could be used to classify lupus using logistic regression.

The first four weeks should include students trying to create as many WIBGIs as possible. At the end of each homework assignment, I added an additional optional section entitle "Project Development" where I reminded the students they need to produce at minimum 3 WIBGI statements.

The main concept here is to encourage exploration and broad thinking. It is okay if students write statements that are not feasible. It is also okay if they write statements that don't conform to the example statement. The goal here is creativity. The idea is to begin generating ideas and determining sources of data; we are trying to encourage data exploration and thinking outside the box from their core studies.