An Introduction to Experimental Design and Linear Models

Andrew P Beckerman (based on text and slides from Mark Rees and Gareth Phoenix)

2021-02-08

Contents

Chapter 1

Introduction

Welcome to An Introduction to Experimental Design and Linear Models

In this mini-module, you'll be learning about the design and analysis of ANOVA and ANCOVA experiments.

This module is compulsory for all, because it forms the foundation for most of the more complex experiments you will do as a researcher. And it is the major step beyond the t-test, 1-way ANOVA, simple regression and chi-square contingency table analyses we covered in semester 1.

If you found Semester 1 ideas hard, please return to the sections on the 1-way ANOVA and the Regression model. Chapter 5 in Getting Started with R (available as an online Resource via STARPlus) covers the material you should know.

You will also need to feel comfortable with dplyr and ggplot - well be reinforcing the old stuff and introducing a few new tricks.

There are 12 Chapters to the book, some only a page or two and some quite a but longer. The majority contain R code that you should recreated in your own scripts as you work along. You should use your scripts to add annotation and notes about the core ideas.

The learning outcome for this mini-module are that you will understand the basic ideas about

- Replication, Randomisation and Reducing Noise
- Experimental Design and Analysis - The Completely Randomised Design
- Experimental Design and Analysis - The Randomised Block Design
- Experimental Design and Analysis - The Latin Square Design
- Experimental Design and Analysis - The ANCOVA Design
- Multiple Regression and Non-linearity.

A note on time allocation: In a 'normal semester', you'd be in the computer room working on these topics for ~12 hours over the course of 3 days in a week.

You should plan on allocating 9-12 hours of your week to this section. You should expect to do the same for the other mini-modules.

1.0.1 The Three Rs

Before we get started, it's vital that you understand that there are some very basic principles needed to ensure that your experiments can provide robust and reliable inference (answers to your questions). The "3 R's".

- **Randomisation**: the random allocation of treatments to the experimental units, to avoid confounding between treatment effects and other unknown effects.
- **Replication**: the repetition of a treatment within an experiment, to quantify the natural variation between experimental units and increase accuracy of estimated effects.
- **Reduce noise:** by controlling as much as possible the conditions in the experiment, e.g. by grouping of similar experimental units in blocks.

As we develop ideas, you should come back to these definitions and see if your understanding of them has improved.

1.0.2 The General Linear Model

This module is focused on a class of model called the General Linear Model. It is not a **GLM**. The **GLM** is a generalised linear model. I know, right?

The general linear model is, as we learned last semester, a model fit in R with the `lm()` function. It includes regression, ANOVA, ANCOVA and variations of these. There are a few key characterstics to remember about these models

The general linear model has the following form

$y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \epsilon$

Where the y is the response variable, the β's are estimated parameters, the X's are the predictor variables and the ϵ comes from a Gaussian distribution with zero mean and constant variance.

Let's decompose that a bit more

There are two types of predictor variable

Metric predictor variables are measurements of some quantity that may help to predict the value of the response. For example, if the response is the blood

pressure of patients in a clinical trial, then age, fat mass and height are potential metric predictor variables. You may know these as **continuous explanatory variables**

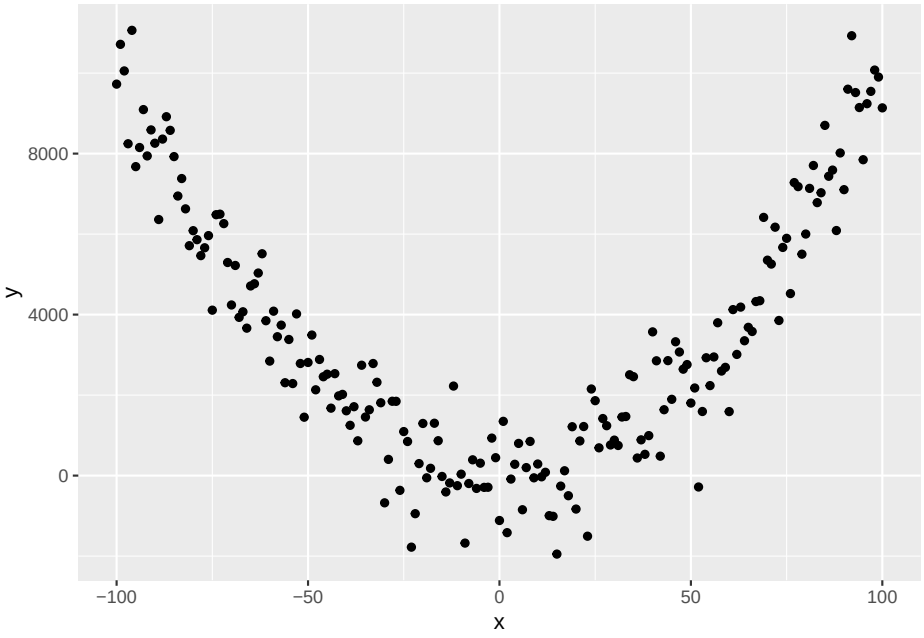
Factor variables are labels that serve to categorize the response measurements into groups, which may have different expected values. Continuing the blood pressure example, factor variables might be sex and drug treatment received (drug A, drug B or placebo, for example). You may also know these as **categorical explanatory variables**.

So, you hopefully can see how this *general* linear model is capable of representing

1. ANOVA – Analysis of variance. Predictors are factors.
2. Regression. Predictor is a metric variable (continuous variable).

3. Multiple regression. Predictors are a metric variables (continuous variables).
4. ANCOVA - Analysis of co-variance. Predictors are a mixture of metric variables (continuous variables) and factors.

Finally, it is important to understand that these data can be modelled with a linear model:



How, you ask!? Well…. consider this equation

$y = 0.01 + x + x^2 + \epsilon$

Referring to our generic model structure above,

$y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \epsilon$

we hopefully can see that $\beta_0 = 0.01$, $\beta_1 = 0$ and $\beta_2 = 1$, where $X_2 = X^2$!

Linear models are perfectly capable of being used to estimate non-linear relationships.

Here is the code to make that figure.

```
# set x range
x <- -100:100
# define y without error
y_det <- 0.01+x^2
# add some random variation
y <- y_det+rnorm(length(x),0,1000)

# create dataframe and plot
df <- data.frame(x, y)
ggplot(df, aes(x = x, y = y))+
  geom_point()
```

1.0.3 Section Readings —-

There are several resources that will help with this section of the stats course, and onwards

- Getting Started with R - An Introducton for Biologists, Second Edition (available as an electronic online resource via StarPlus)
- Experimental Design for the Life Sciences - Nick Colegrave and Graham Ruxton (seen on eBay for £2.50!)
- Of course, the venerable coursebook for APS 240: https://dzchilds.github.io/stats-for-bio/index.html

1.0.4 Install some extra packages —-

In order to make this module more effective, we are going to use some additional resources from CRAN.

Please install these packages, if you have not already, using the install packages tab in RStudio:

- `tidyverse`
- `ggfortify`
- `agricolae`
- `car`

- `gmodels`
- `visreg`
- `patchwork`

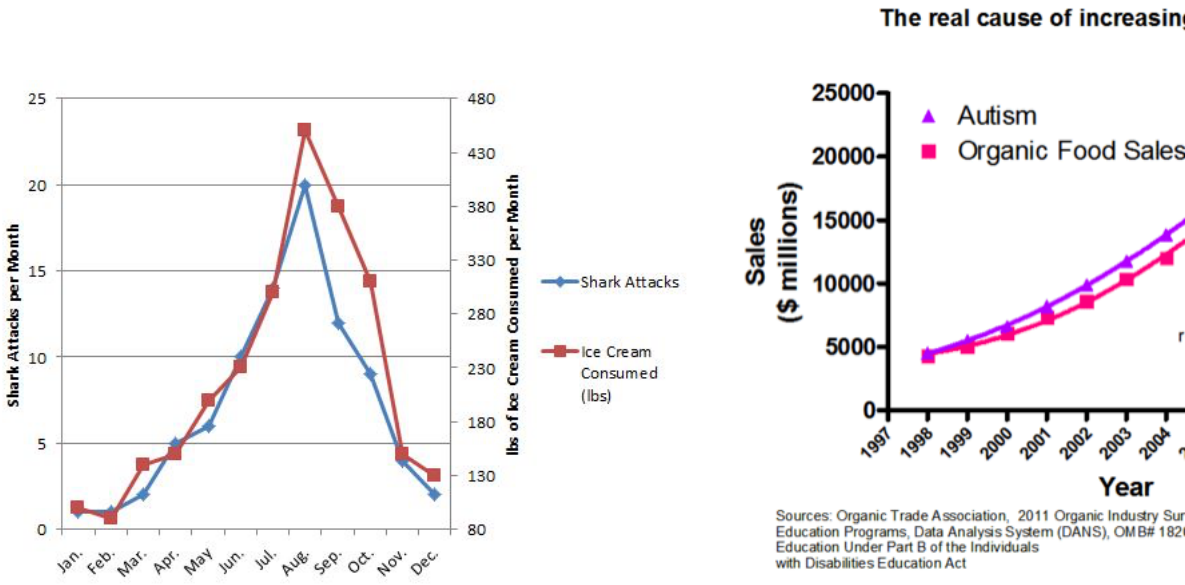
Chapter 2

Introduction To Experiments (Some Philosophy)

Experiments help us answer questions, but there are also non-experimental techniques. What is so special about experiments?

1. Experiments allow us to set up a direct comparison between the treatments of interest.
2. We can design experiments to minimize any bias in the comparison.
3. We can design experiments so that the error in the comparison is small.
4. Most important, we are in control of experiments, and having that control allows us to make stronger inferences about the nature of differences that we see in the experiment. Specifically, we may make inferences about causation.

This last point distinguishes an experiment from an observational study. In an observational study we merely observe which units are in which treatment groups; we don't get to control that assignment. This underpins the classic issue with assigning *causation to correlation* - in the following two examples, there is a strong association between the variables, but there has been no control/manipulation.



Mosteller and Tukey (1977) list three concepts associated with causation and state that two or three are needed to support a causal relationship:

- Consistency – make a change and the response is in the same direction or the amount of response is *consistent* across populations
- Responsiveness – make a change and the response changes according to theory
- Mechanism – make a change and we can monitor/identify a mechanism leading from cause to effect

Let's look at a classic example. Smoking and lung cancer – from 1922 to 1947 annual deaths for lung cancer went from 612 to 9287 (Observation).This was thought in the 1950s to be either an effect of smoking tobacco or atmospheric pollution (Hypothesis). Numerous studies showed that lung cancer was more prevalent in smokers (Observation: consistency). Chemical analyses of tobacco showed it contained carcinogens (Association: mechanism). Public health programs resulted in a reduction in smoking and lung cancer rates decreased (Intervention: responsiveness).

Note the initial study was an observational study and in this case it was not ethical to do the experiment per se!

2.1 How to think about experiments: Components of an Experiment

An experiment has treatments, experimental units, responses, and a method to assign treatments to units. These specify the experimental design.

Not all experimental designs are created equal. A good experimental design must adhere to the 3Rs. It should reveal consistency, responsiveness and mechanism. The way this happens is by avoiding systematic error in measuring things, and allow estimation of error in measurments with precision. In short, a good experimental design must

- Avoid systematic error
- Allow estimation of error
- Be precise
- Have broad validity.

Lets walk through some definitions.

If our experiment has *systematic error*, then our comparisons will be biased, no matter how precise our measurements are or how many experimental units we use. **Randomisation** is our tool to combat *systematic error*.

Even without *systematic error*, there will be random error in the responses - this is what we call variation in what we are measuring or more formally variance. Such variation in responses invariably leads to random error in the treatment comparisons. When we compared two means in the t-test, we had to deal with the variation in both groups!

Experiments are precise when this random error in the treatment comparisons is small. Precision depends on the size of the random errors in the responses, the number of units used (**replication**), and the experimental design used.

Experiments must be designed so that we have an estimate of the size of random error. This permits statistical inference: for example, confidence intervals or tests of significance based on t- or F-statistics.

We cannot do inference without an estimate of this variation. We would like our conclusions to be valid for a wide population, so we need to randomise our subjects or objects we are measuring - for example, we may need to be aware of both sexes and of young and old individuals. But there are always compromises - for example, broadening the scope of validity by using a variety of experimental units may decrease the precision of the responses.

2.1.1 How do we increase precision and reduce bias?

There are several key concepts

2.1.1.1 Blinding

Blinding occurs when the evaluators of a response do not know which treatment was given to which unit. Blinding helps prevent bias in the evaluation, even unconscious bias from well-intentioned evaluators. Double blinding occurs when both the evaluators of the response and the (human subject) experimental units do not know the assignment of treatments to units. Blinding the subjects can also prevent bias, because subject responses can change when subjects have expectations for certain treatments.

2.1.1.2 Placebos

Placebo is a null treatment that is used when the *act* of applying a treatment—any treatment — has an effect. Placebos are often used with human subjects, because people often respond to the process of receiving any treatment: for example, reduction in headache pain when given a sugar pill. Blinding is important when placebos are used with human subjects. Placebos are also useful for nonhuman subjects. The apparatus for spraying a field with a pesticide may compact the soil. Thus we drive the apparatus over the field, without actually spraying, as a placebo treatment.

2.1.1.3 Confounders

Confounding occurs when the effect of one factor or treatment cannot be distinguished from that of another factor or treatment. The two factors or treatments are said to be confounded. Except in very special circumstances, confounding should be avoided. Consider the following example. We plant corn variety A in Yorkshire and corn variety B in Lancashire. In this experiment, we cannot distinguish location effects (Yorkshire vs. Lancashire) from variety effects (cornA vs. cornB) — the variety factor and the location factor are confounded.

This is despite the fact that we know that Yorkshire will be better…. (that's a joke)

2.2 Experimental vs Measurement units

A common source of difficulty in designing experiments is the distinction between experimental units and measurement units. We need to know the experimental units, as this is the key value used to generate our inference.

Consider an educational study, with six classrooms of 25 pupils. Each classroom of students is then assigned, at random, to one of two different reading programmes.

At the end of a 6 week term, all the students are evaluated via a common reading exam.

So, are there six experimental units (the classrooms) or 150 (25*6; the students)? We measured the reading ability of the students… but they were in classroom sets of 25….

2.2.0.1 Identifying the experimental unit - an example of pseudo-replication

To identify the experimental units the key question is: What did we randomly allocate treatments to?

If we randomly allocated reading programmes to students, then students would be the experimental units. But we don't so the classroom is the experimental unit – it is the classroom to which we randomly allocated treatments. The classroom is the experimental unit. But we don't measure how a classroom reads; we measure how students read. Thus students are the measurement units for this experiment.

Confusing these two things can lead to **pseudo-replication**. Treating measurement units as experimental usually leads to overoptimistic analysis — we will reject null hypotheses more often than we should, and our confidence intervals will be too narrow. The usual way around this is to determine a single response for each experimental unit.

2.2.0.2 This really matters.

Consider an experiment with 2 growth chambers each containing 100 plants. One of the chambers received enhanced C02. One night after collecting data you leave the door open on the C02 chamber and the temperature drops and so the plants grow more slowly. When you come to analyze the data you get a highly significant effect but C02 results in reduced plant growth not what you expect.

This is entirely plausible.

Consider a second experiment where you really do have 200 growth chambers and randomly allocate plants to each. If you forget to close one door it really has no effect as just one plant is affected.

To get the same effect as in the first experiment you would have to accidentally leave the doors open on all the C02 chambers. This is very unlikely indeed!!!

There are 9x1058 ways selecting 100 chambers from 200 chambers so the chance of accidentally picking all the C02 chambers is 1/ 9x1058 0 (stars in universe 7 x 1022).

Proper **randomization** and **replication** is very different from **pseudo-replication**.

2.2.1 Take Home Message: Randomization properly with Replication protects against Confounding

> An experiment is randomized if the method for assigning treatments to units involves a known, well-understood probabilistic scheme. The probabilistic scheme is called a randomization.
>
> In general, more experimental units with fewer measurement units per experimental unit works better. However, smaller experimental units are inclined to have greater edge effect problems than are larger units, so this recommendation needs to be moderated by consideration of the actual units.

No matter which features of the population of experimental units are associated with our response, our randomizations put approximately *half the individuals with these features* into *each treatment group*.

Recall our example above of considering sex and age of subjects and imagine a treatment with two levels (hot and cold). Done well, proper randomisation will put approximately half the males, half the females, half the older, half the youger etc into each of the treatment levels.

The beauty of randomization is that it helps prevent confounding, even for factors that we do not know are important.

2.2.2 A non-randomized experiment – haphazard is NOT randomized

A company is evaluating two different word processing packages for use by its clerical staff. Part of the evaluation is how quickly a test document can be entered correctly using the two programs. We have 20 test secretaries, and each secretary will enter the document twice, using each program once.

Suppose that all secretaries did the evaluation in the order A first and B second. Does the second program have an advantage because the secretary will be familiar with the document and thus enter it faster? Or maybe the second program will be at a disadvantage because the secretary will be tired and thus slower.

Randomization generally costs little in time and trouble, but it can save us from `disaster`.

Anything that might affect your responses should be randomized! For example

- If the experimental units are not used simultaneously, you can randomize the order in which they are used.
- If the experimental units are not used at the same location, you can randomize the locations at which they are used.
- If you use more than one measuring instrument for determining response, you can randomize which units are measured on which instruments.

2.3 Mini-Quiz

A PhD student want to determine the effects of protein on beetle fecundity so they design an experiment with a control and protein enhanced diet. To assign beetles to each of the treatments they pour a culture onto the table and catch the first 30 beetles that run to the edge of the table, these receive the protein enhanced diet. The next 30 beetles go in the control. **Is this randomized?**

2.4 Replication: How many?

No simple rules… it depends… on….

- Resources available
- Variability of experimental units
- Treatment structure – more later
- Size of effect that's important
- Relative importance of different comparisons

There is, however, a set of tools that can help with estimating sample sizes. It's called power analysis and requires that you have some a priori estimate of the expected variation in your response variable.

Chapter 3

Standard Errors, Precision and Sampling

3.1 Standard Errors and precision

One of the most important statistical metrics to wrap your heads around is the standard error of the mean.

We introduced this in the first semester and there is a good discussion of it in the APS 240 reading. You should re-read this section:

https://dzchilds.github.io/stats-for-bio/sampling-error.html

The take home message is that the standard error is a quantitative measure of sampling error variation. It is also very sensitive to the sample size, because the equation is `sd(variable)/sqrt(sample_size)`. As the sample size increases, the standard error, an estimate of the sampling error, goes down.

$$\text{Standard Error mean} = \frac{\sigma}{\sqrt{n}}$$

Sample size ≈ 30 gives big reduction in SE

3.1.1 SE of a Difference and Balance.

Interestingly, we often are estimating the standard error of a difference. Remember the t-test? The test statstic is all about the difference between the means. We need a precise estimate of that difference to assess whether the null hypothesis (the differene = 0) can be rejected or not.

The standard error of a difference behaves differently, however.

Standard Error of difference = $\sigma\sqrt{\frac{1}{n_1} + -$

If you have 100 experimental units how should they be divided between 2 treatments?

What this suggests is that we allocate our sample sizes in a balanced manner - 50% to each group/treatment level. In general balanced designs are good – equal numbers of replicates per treatment.

3.1.2 However sometimes unbalanced designs are better.

Say we have g treatments one of which is a control, say treatment 1, and we want to compare each of the other treatments with the control, so treatment 2 vs Control, treatment 3 vs Control, treatment 4 vs Control, etc. Then as we compare everything with the control we want more replicates of the control. The best allocation is $n_c = n_t\sqrt{(g-1)}$ where n_t is the number of replicates in the non-control treatment.

3.2 The CRD: Completely Randomised Design

Against this backdrop of understanding **randomisation** and **replication**, the most basic design is the Completely Randomized Design:

We have g treatment levels to compare (e.g. Hot, Warm, Cold) and N units to use in our experiment. For a completely randomized design:

1. Select sample sizes $n_1, n_2, ..., n_g = N$. Often we set the n's to be equal so we have a balanced design.
2. Choose n_1 units at random to receive treatment 1, n_2 units at random from the N − n1 remaining to receive treatment 2, and so on.

Chapter 4

Examples and Challenge Questions

In this section, we will review a field based experimental design example. There are challenge questions to answer.

We will also introduce tools to generate randomised experimental designs - this is a good trick to have up your sleeves!

4.1 Designing your first experiment

You are challenged to design an Arctic field manipulation experiment to evalauate UV-B radiation and increased CO2 impacts on plant growth.

Context: an arctic tundra study Increasing ultraviolet-B (+UV-B) radiation from ozone depletion (the arctic ozone hole) Increasing atmospheric CO2 (+CO2) from anthropogenic emissions For plants: +UV-B potentially harmful, +CO2 potentially beneficial

Hypotheses Elevated (+) UV-B radiation will reduce the growth of arctic plants Elevated (+) CO_2 will increase the growth of arctic plants

The resources available to you are constrained. The arctic research station has given permission for 16 plots (each 2m x 2m) in the natural vegetation nearby.

One +UV-B plot (2m x 2m) costs £4000 (this provides the UV-B lamps, frame, power and control system, wooden walkways around the plots) One $+CO_2$ plot (2m x 2m) costs £6000 (this provides a CO2 release system, CO2 control and covers CO2 purchase costs, wooden walkways around the plots) One control plot (2m x 2m) costs £200 (marking posts, wooden walkways around the plots)

You have a budget of £61,000

Design an experiment to test the hypotheses (i.e. the design of the plots and treatments including replication, not what measurements you will take - which will be plant growth rates….). Think hard about this. How many treatments do you have? How many plots/treatment would you like to allocate? Is this

possible? Will this be a balanced design, given your max budget? If it isn't, what rule can you use to allocate them replicates to treatments?

Write an answer down, before moving to the next section. We'll provide the answer separately!

4.2 Using R to generate Experimental Designs

This is the place to stop for a moment and organise your R-course life. My suggestion is to do the following.

Create an Experimental Design folder on your computer. Put a data folder inside that. Use RStudio to create an RStudio Project file inside the Experimental Design folder. This will be the only place you need to focus for this mini-module.

Now, let's set up a script to do build this experimental design and actually generate soome data and do an analysis.

As with every script last semester, we start with a preamble and some libraries

```
# Designing experiments with R: and introduction
# MY NAME
# DATE

#libraries I need

library(tidyverse)
library(ggfortify)
library(agricolae)
library(car)
library(gmodels)
library(visreg)
```

That's alot of packages, but you'll see how we use them for designing and then analysing data.

To design and experiment, we are going to use the `design.csd` function from the *agricolae* package, which we used last term to help with the tukey tests. It does WAY MORE!

This package has a great online help resource: https://myaseen208.github.io/agricolae/articles/Intro_agricolae.html

```
# use this to make sure the random allocation pattern created for the design is the same for you
set.seed(123)
```

```
# define the treatments
treatments <- c("Control","UVB","CO2")

# define the design (this does some randomisation for you!)
design <- design.crd(treatments, r = c(6,5,5), serie = 0)

# view the design - note that the actual data are in $book
glimpse(design)
```

```
## List of 2
##  $ parameters:List of 7
##   ..$ design: chr "crd"
##   ..$ trt   : chr [1:3] "Control" "UVB" "CO2"
##   ..$ r     : num [1:3] 5 6 5
##   ..$ serie : num 0
##   ..$ seed  : int 515190382
##   ..$ kinds : chr "Super-Duper"
##   ..$       : logi TRUE
##  $ book      :'data.frame':  16 obs. of  3 variables:
##   ..$ plots     : num [1:16] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ r         : int [1:16] 1 1 1 2 2 2 3 3 4 3 ...
##   ..$ treatments: chr [1:16] "UVB" "CO2" "Control" "UVB" ...
```

```
# grab that book
use_design <- design$book

# show the design
use_design
```

```
##    plots r treatments
## 1      1 1        UVB
## 2      2 1        CO2
## 3      3 1    Control
## 4      4 2        UVB
## 5      5 2        CO2
## 6      6 2    Control
## 7      7 3        CO2
## 8      8 3        UVB
## 9      9 4        CO2
## 10    10 3    Control
## 11    11 4        UVB
## 12    12 4    Control
## 13    13 5    Control
## 14    14 6    Control
```

```
## 15    15 5        CO2
## 16    16 5        UVB
```

Great scott! So, we have 16 plots and we've randomly allocated the set of treatments to these plots, with 6 controls, and 5 of the non-control treatments. This is just what we wanted.

Time to move on to the next chapter, but stick with this script! We will come back to this example later.

Chapter 5

Design and Analysis of Experiments

In this section, we will design and experiment, make up data for the experiment, and analyse the experiment.

This is a very good process to go through - you will understand more fully how to generate randomisation in your experiments, and you will better understand how data that you collect is *structured* - from where does variation come in your measurements.

Another way to think about this is to consider that we are learning how statistics looks at data you've collected. We are going to make up data, where we know the answer to the question. Then we are going to plot these data, and analyse them.

5.1 Example one

We are now going to cheat by generating artificial data we we don't have time to go run the experiment.

The experiment is about plant biomass yield under several herbicide treatments: a control and two herbicides, and a third treatment that is a placebo - applied water but no herbicide. The hypothesis is....?

Start a new section of your script (some hashes) called Herbicide Example.

Now, enter the following code in your script file, then run it.

```
# set the random seed - this will ensure that your results and mine here are the same.
set.seed(123)

#treatment names we have a control and two herbicides, Herb3 is a placebo (applied wat
treat <- c("Cont","Herb1","Herb2","Placebo")

#number of replicates
Nreps <- 30

#Total number of experimental units
Total.units <- Nreps * length(treat)

#Our completely randomized design
# not the trick of adding the $book at the end of the code
design <- design.crd(treat, Nreps, serie = 0)$book

# check it out (the first 10 rows)
head(design, 10)
```

```
##    plots r   treat
## 1      1 1    Cont
## 2      2 1   Herb1
## 3      3 2   Herb1
## 4      4 1 Placebo
## 5      5 2 Placebo
## 6      6 3 Placebo
## 7      7 3   Herb1
## 8      8 2    Cont
## 9      9 1   Herb2
## 10    10 2   Herb2
```

5.1.1 Making up data

Right, lets make up some artificial data so we know the right answers – this is a very good way of checking you understand what's going on.

First, lets define some randon variation centred around 0 with a standard deviation of 3:

```
#Our experimental errors, normal distribution mean = 0, standard deviation = 3
# rnorm is random normal distribution - the bell curve!
error <- rnorm(Total.units, mean=0, sd=3)
```

Now we need to actually generate a response variable. The thing we 'measured'.

For this experiment we are *measuring* yield - the grams of dry biomass at the end of the experimental season.

To do this, we have to think about the mean of the yield (we'll set it to 20), the deviation caused by Herbicide 1 (+5 average yield), the deviation caused by Herbicide 2 (+6 average yield) and the deviation caused by the placebo (nothing) and the error among observations (error). If this works, there should be Controls and Placebos with values around 20, and Herbicide values around ±25-26. But keep in mind… we have a standard deviation of 3 in the error… what will that do? Let's see.

```
#the observations of yield
design$obs <-
  # mean yield Control
  20 +
  # deviation caused by Herbicide 1
  (design$treat=="Herb1") * 5 +
  (design$treat == "Herb2") * 6 +
  (design$treat == "Placebo")*1 +
  error

# look at it
design$obs
```

```
##   [1] 25.56702 23.32992 23.39985 21.54529 20.96267 22.34314 24.19572 15.45247
##   [9] 28.25849 28.09586 25.03735 26.36961 20.44643 16.30186 25.99108 29.62076
##  [17] 24.05645 22.01011 25.48992 29.60826 25.83590 23.77995 21.06598 21.85137
##  [25] 19.10903 20.54631 21.71607 24.78175 15.84089 24.56500 26.21161 19.96028
##  [33] 16.21851 24.50682 17.80657 23.74981 30.64518 24.90522 27.33204 18.15174
##  [41] 13.41104 27.87253 19.64131 22.48390 25.17527 15.41037 16.58438 15.85613
##  [49] 20.56557 20.75066 19.84848 19.97868 22.47068 25.11759 25.55048 24.92058
##  [57] 18.45873 22.02616 23.09377 15.56172 26.78291 18.20024 21.20411 25.58635
##  [65] 25.67710 26.29930 22.20238 20.63010 30.71839 18.37104 29.64641 14.20080
##  [73] 30.80997 22.67665 29.92778 20.63348 21.45997 20.32328 25.25901 18.08680
##  [81] 18.75511 24.83941 27.00049 23.06356 27.87863 18.63813 16.75083 24.45870
##  [89] 19.76979 23.40537 21.49970 21.09057 18.48304 18.64256 23.66625 28.52046
##  [97] 18.82317 29.38274 22.55174 26.41338 25.95806 24.86078 21.90836 20.61218
## [105] 25.47680 21.99004 23.06115 20.40581 15.70540 23.30232 28.04868 18.26389
## [113] 30.76203 24.50311 19.17553 25.11028 26.65809 18.84203 26.01325 26.79324
```

```
# look at the design now.
head(design)
```

```
##   plots r   treat      obs
## 1     1 1    Cont 25.56702
```

```
## 2     2 1   Herb1 23.32992
## 3     3 2   Herb1 23.39985
## 4     4 1 Placebo 21.54529
## 5     5 2 Placebo 20.96267
## 6     6 3 Placebo 22.34314
```

5.1.2 Into the dplyr and ggplot pipeline.

Great stuff. Now we can move to our standard data management and visualisation pipeline.

1. review the data
2. sumamrise the data with dplyr - generate means and se's for the treatments
3. visualise with ggplot2

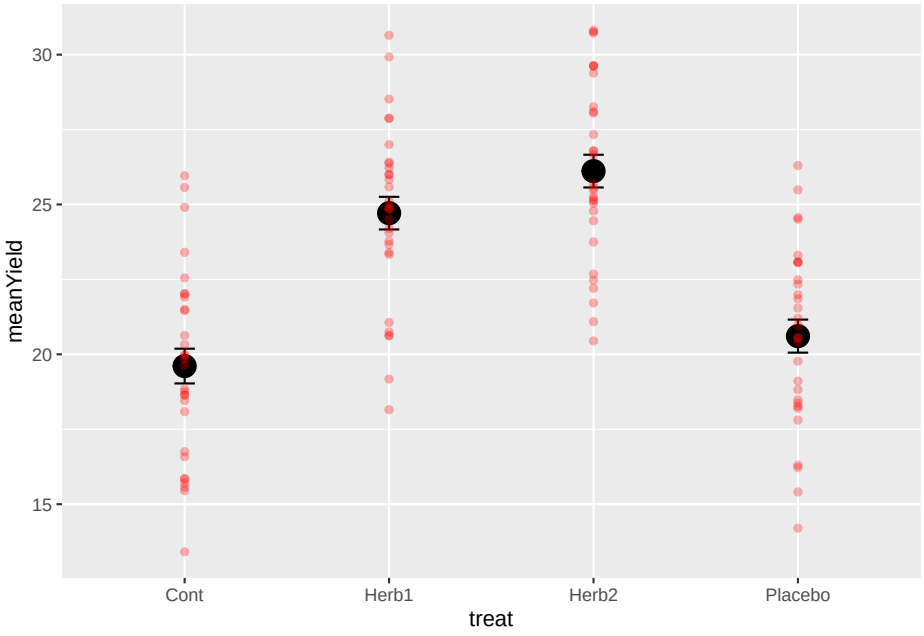
```
# check the data
glimpse(design)
```

```
## Rows: 120
## Columns: 4
## $ plots <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18...
## $ r     <int> 1, 1, 2, 1, 2, 3, 3, 2, 1, 2, 3, 4, 4, 4, 5, 5, 6, 3, 5, 6, 7...
## $ treat <chr> "Cont", "Herb1", "Herb1", "Placebo", "Placebo", "Placebo", "H...
## $ obs   <dbl> 25.56702, 23.32992, 23.39985, 21.54529, 20.96267, 22.34314, 2...
```

```
# summarise to get means and ses
sumDat <- design %>%
  group_by(treat) %>%
  summarise(
    meanYield = mean(obs),
    seYield = sd(obs)/sqrt(n())
  )
```

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

```
# plot the raw data and the mean±se
# start with the mean±se and then add the raw data
ggplot(sumDat, aes(x = treat, y = meanYield))+
  geom_point(size = 5)+
  geom_errorbar(data = sumDat, aes(ymin = meanYield - seYield, ymax = meanYield+seYield
                width = 0.1)+
  geom_point(data = design, aes(x = treat, y = obs), colour = 'red', alpha = 0.3)
```



A few things to notice.

1. The data are quite variable and the means of the herbicide treatments are roughly 5 and 6 units higher. This is as we expected….
2. The standard errors are quite small! Why is that!?
3. For those of you interested in some extra reading and thinking, the 95% Confidence Interval around the means can be calculated using `1.96*SE` == `1.96*sd(obs)/sqrt(n())`. Go ahead and do that and look into that if you want…

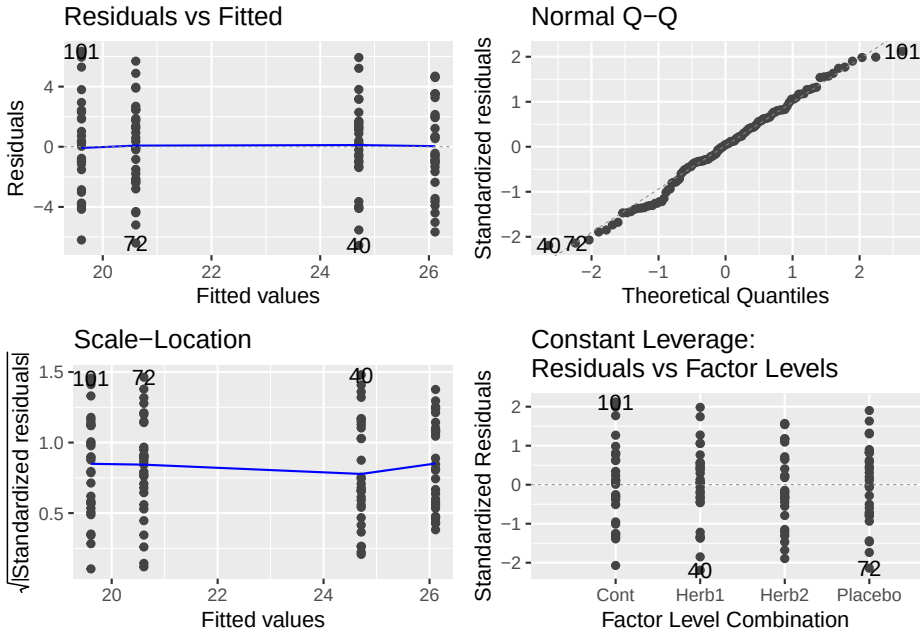
5.2 The One-Way ANOVA.

If you've been paying attention, we've essentially designed and plotted the data for a 1-way ANOVA. These data are very similar to the daphnia parasite data we finished semester 1 with.

To analyse these data, we use the `lm()` function to build the model, check assumptions, and then make inference. Let's go.

```
# the model
modYield <- lm(obs ~ treat, data = design)

# assumptions
autoplot(modYield)
```




```
# inference: anova
anova(modYield)
```

```
## Analysis of Variance Table
##
## Response: obs
##            Df  Sum Sq Mean Sq F value    Pr(>F)
## treat       3  888.39 296.129  31.968 3.905e-15 ***
## Residuals 116 1074.54   9.263
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# contrasts
summary(modYield)
```

```
##
## Call:
## lm(formula = obs ~ treat, data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5589 -1.8698  0.1395  2.0230  6.3494
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   19.6087     0.5557  35.288  < 2e-16 ***
## treatHerb1     5.1019     0.7858   6.492 2.18e-09 ***
## treatHerb2     6.5048     0.7858   8.277 2.43e-13 ***
## treatPlacebo   0.9993     0.7858   1.272    0.206
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.044 on 116 degrees of freedom
## Multiple R-squared:  0.4526, Adjusted R-squared:  0.4384
## F-statistic: 31.97 on 3 and 116 DF,  p-value: 3.905e-15
```

5.2.1 Making insight and inference

Lets walks through things very discretely.

1. Our graph suggests that herbicide treatments have an effect of increasing yield.
2. Our model is designed to test this hypothesis - are any of the differences among means non-zero?
3. Our hypothesis is probably really about whether the herbicide and placebos are differnt than the controls
4. Our diagnostics are fantistic… the best you've ever seen.
5. The Anova Table confirms that there are differences - we can reject the null hypothesis
6. The summary table confirms that Herb1 and Herb2 are both larger than controls and the Placebo is not.

How do we interpret even more? The estimate associated with Control is 20! Just where it should be.

The estimates associated with Herb1, Herb2 and Placebo are the differences between the mean of these treatments and the control (the reference level!). These differences are positive for Herb1 and Herb2, close to 5 and 6 respectively (as expected) and this positive difference is not 0 via the statistical test.

However, the differnce for Placebo is close to 0 and therefore we can not reject the null hypothesis test that it differs from control. GENUIS!

5.3 A priori vs. Post-Hoc Contrasts

In the semester 1, we introduced how to do a Tukey Test.

This is known as an *a posteriori* test – testing the significance of things suggested by the experiment, also known as data snooping or data dredging. These

are multiple comparison methods (Bonferroni, Scheffe method, Tukey honest significant difference, Duncan's multiple range test) which try to control the chance of getting a significant result by chance.

To understand the risks of these, consider this experimental design. We have 7 treatments. With 7 treatments, there are 21 pairwise comparisons. With p-value threshold of 0.05 we expect 1/20 (5/100) tests to be significant. So with this 7 treatment and 21 comparison design, would you expect a signficant result by chance? You betyja.

This is why, unless a priori (in advance) you can justify ALL pairwise comparisons, a tukey test may not be appropriate.

Some statisticians really don't like them "In my view multiple comparison methods have no place at all in the interpretation of data" Nelder (very well respected statistician).

5.3.1 The more appropriate approach.

The classical approach is to specify a priori (before experiment) a set of hypotheses then test them using contrasts. For our experiment, as noted above, we were probably interested in what our treatment contrasts provided - tests of difference with the control.

Specifying specific contrasts is easy once you get your head around the 'structure' of the syntax. Lets have a go with specifying a comparison JUST between Herbicide 1 and the control.

```
# check the levels and ORDERING of the treatments
levels(design$treat)
```

```
## NULL
```

```
# define the contrast you want using -1, 1 and 0's
# this says compare control with herbicide 1.... and ignore the Herb2 and Placebo
# we give the reference -1 and the 'other' 1.
contrast <- c(-1,1,0,0)

# use the fit.contrast function from gmodels
fit.contrast(modYield, "treat", contrast)
```

```
##                     Estimate Std. Error  t value     Pr(>|t|)
## treat c=( -1 1 0 0 ) 5.101935  0.7858436 6.492304 2.181992e-09
## attr(,"class")
## [1] "fit_contrast"
```

```
# remind ourselves of the contrast from the summary table
summary(modYield)
```

```
##
## Call:
## lm(formula = obs ~ treat, data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5589 -1.8698  0.1395  2.0230  6.3494
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   19.6087     0.5557  35.288  < 2e-16 ***
## treatHerb1     5.1019     0.7858   6.492 2.18e-09 ***
## treatHerb2     6.5048     0.7858   8.277 2.43e-13 ***
## treatPlacebo   0.9993     0.7858   1.272    0.206
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.044 on 116 degrees of freedom
## Multiple R-squared:  0.4526, Adjusted R-squared:  0.4384
## F-statistic: 31.97 on 3 and 116 DF,  p-value: 3.905e-15
```

Notice that the results are the same from the summary(modYield) and the fit.contrast.

If we want to compare the two herbicides we can use this approach. Note in advance that this contrast DOES NOT exist in the summary table!

```
# define the contrast you want using -1, 1 and 0's
# this says compare herb1 with herb2, ignoring the control and placebo.
# we give the reference -1 and the 'other' 1.
contrast <- c(0,-1,1,0)

# use the fit.contrast function from gmodels
fit.contrast(modYield, "treat", contrast)
```

```
##                      Estimate Std. Error  t value   Pr(>|t|)
## treat c=( 0 -1 1 0 ) 1.402879  0.7858436 1.785189 0.07684408
## attr(,"class")
## [1] "fit_contrast"
```

Isn't this cool? Note that the difference reported is the difference between the two means:

```
# check our summary data
sumDat
```

```
## # A tibble: 4 x 3
##   treat   meanYield seYield
##   <chr>       <dbl>   <dbl>
## 1 Cont         19.6   0.580
## 2 Herb1        24.7   0.545
## 3 Herb2        26.1   0.546
## 4 Placebo      20.6   0.552
```

Here it is: $26.2 - 24.2 = 2$

This says that despite the difference we created of ~1 unit of yield between Herb1 and Herb2, when we defined error and generated the data, it created a significant difference detectable with statistics.

5.3.2 comparing the average of the herbicide effect with the control.

This might be a comparison you intended to make also… the average effect of herbicides in general. To do this, we expand the idea of -1,1 and 0's to include 1/2s (and yes, 1/3's and more are possible):

```
# define the contrast you want using -1, 1 and 0's
# this says compare control with the average of herbicide 1 and 2, ignoring the placeb
# we give the reference -1 and the 'other two' a 1/2 each.
contrast <- c(-1,1/2,1/2,0)

# use the fit.contrast function from gmodels
fit.contrast(modYield, "treat", contrast)
```

```
##                          Estimate Std. Error  t value     Pr(>|t|)
## treat c=( -1 0.5 0.5 0 ) 5.803375  0.6805605 8.527346 6.484593e-14
## attr(,"class")
## [1] "fit_contrast"
```

Again, checking sumDat

```
sumDat
```

```
## # A tibble: 4 x 3
##   treat   meanYield seYield
```

```
##   <chr>       <dbl>   <dbl>
## 1 Cont         19.6   0.580
## 2 Herb1        24.7   0.545
## 3 Herb2        26.1   0.546
## 4 Placebo      20.6   0.552
```

$(24.2 + 26.2)/2 = 25.2$ –> $25.2 - 20 = 5.2$

5.3.3 writing this up...

Fill in these blanks using the various contrasts you made above!

> We conclude that herbicides on average cause an _____ gram increase in yield (t = _____ , p = _____). We also note that there was a significant difference of _____ grams between the herbicides (t = _____ p = _____). The additional placebo treatment had no effect on yield (t = _____ p = _____).

Chapter 6

Randomized Complete Block design (RCB)

Blocking allows us to reduce the experimental error.

A block is a group of experimental units that are homogeneous in some sense – in the same place, or measured at the same time, or by the same person. So when constructing blocks we try and select experimental units that are homogeneous within blocks but units in different blocks may be dissimilar.

Why block? When we use a completely randomised design, the location or timing of our treatment 'plots' (patches, incubators, locations in a 96 well plate) can generate heterogeneity in experimental error (variation). As the variance of the Experimental Error increases, confidence intervals get wider and the power of our analysis decreases - it's harder to detect effects of our treatments against the background noise. Ideally we would like to use experimental units that are homogeneous so the experimental error will be small. Blocking does this.

The simplest blocked design is the **Randomized Complete Block design (RCB)**

We have one complete set of treatments in each block. Say you have g Treatments and r Blocks then the total number of experimental units is?

In the first block, we randomly assign the g treatments to the n units; we do an independent randomization, assigning treatments to units in each of the other blocks. This is the RCB design.

For example, consider the following matrix: the rows are the blocks, the letters the different treatments. In each block, each treatment is represented, but it is in a different location in the block (randomisation of the g treatments in the n units). The blocks are in a sequence - left to right - this could be different days,

different locations or different positions on a hillside, for example representing an elevation or soil moisture gradient.

The Blocks are designed to 'capture' that underlying source of variability and allow us to detect among treatment differences more effectively.

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,] "A"  "B"  "A"  "E"  "D"
## [2,] "C"  "A"  "D"  "C"  "A"
## [3,] "D"  "E"  "B"  "D"  "C"
## [4,] "E"  "C"  "E"  "B"  "E"
## [5,] "B"  "D"  "C"  "A"  "B"
```

It is important to note that blocks exist at the time of the randomization of treatments to units. We cannot impose blocking structure on a completely randomized design after the fact; either the randomization was blocked or it was not.

We use an RCB to increase the power and precision of an experiment by decreasing the error variance. This decrease in error variance is achieved by finding groups of units that are homogeneous (blocks) and, in effect, repeating the experiment independently in the different blocks. The RCB is an effective design when there is a single source of extraneous variation in the responses that we can identify ahead of time and use to partition the units into blocks.

In short ALWAYS block your experiment, if you can.

You can have spatial blocks, or temporal blocks where you repeat the experiment at different times, or block by batch.

In general, any source of variation that you think may influence the response and which can be identified prior to the experiment is a candidate for blocking.

6.1 An example of the RCBD

Lets modify our previous example to including blocking. Start another section with some ## and call it Blocking Example.

Of course, if you want to start another script, you can, but make sure you include all of the library()'s again!

```
#Randomised Complete Block Design

# ensure allocation is the same
set.seed(123)
```

```
# define the treatments
treat <- c("Control","Herb1","Herb2","Placebo")

# define the number of blocks
Nblocks <- 5

# consider this
Total.units <- Nblocks * length(treat)

# build the design
design <- design.rcbd(treat, Nblocks, serie = 0)$book

# look at it
design
```

```
##    plots block   treat
## 1     11     1 Placebo
## 2     12     1 Control
## 3     13     1   Herb2
## 4     14     1   Herb1
## 5     21     2 Control
## 6     22     2   Herb1
## 7     23     2   Herb2
## 8     24     2 Placebo
## 9     31     3   Herb2
## 10    32     3   Herb1
## 11    33     3 Control
## 12    34     3 Placebo
## 13    41     4 Control
## 14    42     4   Herb2
## 15    43     4 Placebo
## 16    44     4   Herb1
## 17    51     5 Placebo
## 18    52     5 Control
## 19    53     5   Herb1
## 20    54     5   Herb2
```

This is like the matrix above, but in tidy format! Excellent. This is how a block design looks in tidy-land.

Now, lets generate some data again.

```
# set seed again ...
set.seed(123)
```

```
# define the error - note how we use the variable Total.units to get the number of obs
error <- rnorm(Total.units, mean = 0, sd = 1) # is this more or less variation than be

# generate the observations
# note that we are now generating larger differences (10 and 9) among treatments
# e.g. Herb1 is 10 units larger than the control.
design$obs <- 20 +
  (design$treat=="Herb1") * 10 +
  (design$treat == "Herb2") * 9 +
  (design$treat == "Placebo") * 1 +
  # note that we are defining variation among blocks here
  # block 1 is on average 10 units higher.... and block 5 is...
  (design$block==1) * 10 - (design$block==5) * 10 +
  error

head(design, 10)
```

```
##    plots block   treat      obs
## 1     11     1 Placebo 31.18707
## 2     12     1 Control 31.99603
## 3     13     1   Herb2 41.29937
## 4     14     1   Herb1 41.12746
## 5     21     2 Control 21.08111
## 6     22     2   Herb1 30.56917
## 7     23     2   Herb2 28.09290
## 8     24     2 Placebo 21.02990
## 9     31     3   Herb2 27.44097
## 10    32     3   Herb1 30.29749
```

6.2 Analysing the CRBD

I'll leave it to you now to generate the following picture of the means ± standard errors.

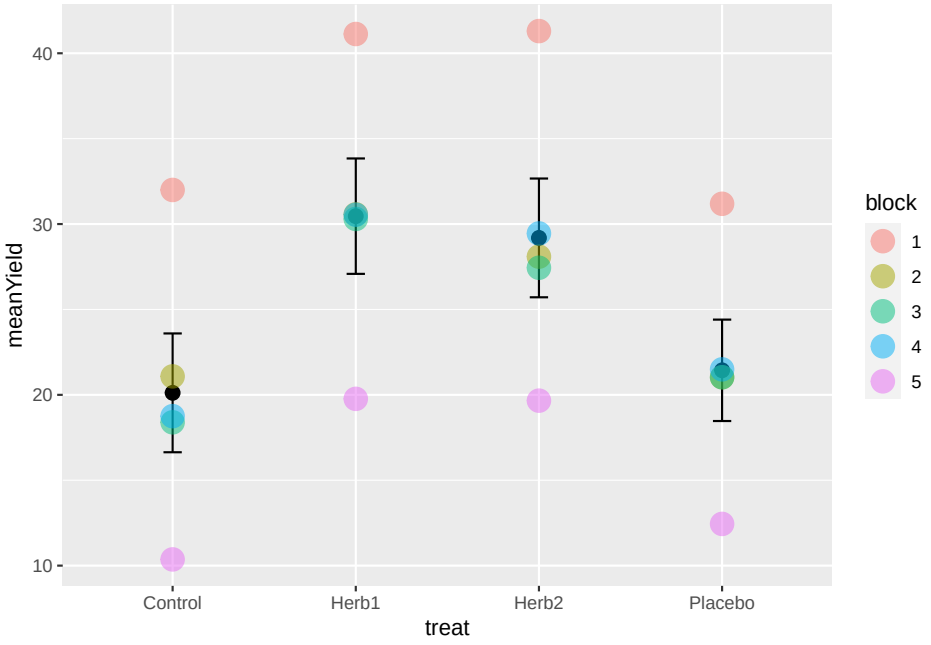
This requires thinking hard about the use of dplyr tools (`group_by()` and `summarise()`) and ggplot (adding more than one layer from two different sources of data - the summary data and the raw data).

Can you see our generation of block 1 and 5's variation?

```
## Rows: 20
## Columns: 4
## $ plots <dbl> 11, 12, 13, 14, 21, 22, 23, 24, 31, 32, 33, 34, 41, 42, 43, 4...
## $ block <fct> 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5
```

```
## $ treat <fct> Placebo, Control, Herb2, Herb1, Control, Herb1, Herb2, Placeb...
## $ obs   <dbl> 31.18707, 31.99603, 41.29937, 41.12746, 21.08111, 30.56917, 2...
```

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



6.2.1 Building the model

In order to understand what's going on with blocking, and it's importance, lets build the naive model that ignores block - treating this as a CRB - and the correct model, letting block absorb some of the variation.

```
# models
naive_model <- lm(obs ~ treat, design)
block_model <- lm(obs ~ block + treat, design) # note the order is important

# anova tables
anova(naive_model)
```

```
## Analysis of Variance Table
##
## Response: obs
##           Df Sum Sq Mean Sq F value  Pr(>F)
## treat      3 417.82 139.274  2.5085 0.09579 .
```

```
## Residuals 16 888.34  55.521
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(block_model)
```

```
## Analysis of Variance Table
##
## Response: obs
##           Df Sum Sq Mean Sq F value    Pr(>F)
## block      4 877.08 219.270  233.68 2.871e-11 ***
## treat      3 417.82 139.274  148.43 9.469e-10 ***
## Residuals 12  11.26   0.938
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The first important thing to focus on here is the difference in the Mean Sq Residual Errors - in the `naive_model`, it is 55.43. In the `block_model`, it is 0.86.

The second important thing to notice is that haveing allocated variation to block in the `block_model`, and thus reducing the error variation, the *treatment* effect shifts from being insignificant to significant.

6.2.2 Are the estimates of the parameters what we expect?

Lets check that the model is estimating differences as we might have expected. We can do this using the summary table.

Let's remember that, for example, the mean of Herb1 is expected to be 10 units higher than control with a yield of 20, and block 1 is supposed to be ~10 units higher than 2,3,4.

```
summary(block_model)
```

```
##
## Call:
## lm(formula = obs ~ block + treat, data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3505 -0.7196  0.2147  0.6396  1.0719
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   31.2184     0.6126  50.957 2.14e-15 ***
## block2       -11.2092     0.6850 -16.365 1.43e-09 ***
## block3       -12.1132     0.6850 -17.685 5.84e-10 ***
## block4       -11.3415     0.6850 -16.558 1.25e-09 ***
## block5       -20.8449     0.6850 -30.433 9.94e-13 ***
## treatHerb1    10.3450     0.6126  16.886 9.96e-10 ***
## treatHerb2     9.0721     0.6126  14.808 4.50e-09 ***
## treatPlacebo   1.3192     0.6126   2.153   0.0523 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9687 on 12 degrees of freedom
## Multiple R-squared:  0.9914, Adjusted R-squared:  0.9864
## F-statistic: 197.1 on 7 and 12 DF,  p-value: 2.009e-11
```

In this table, the *INTERCEPT* is specifying the **FIRST BLOCK** and the **CONTROL TREATMENT LEVEL** - we know this because it's these words that are missing from the rest of the table, and they are each the first alphanumerially in the list of blocks and treatments.

The value of the control, block 1 is approximately 30! Which is 20+10, which is what we expected. The value of Herb1 is ~10 units higher than this (remember, the value 9.84 is the DIFFERENCE between the control and treatment) And the value of block 5 is reported as 20 unites lower than block 1 control. This too is correct because, as above, block 1 control is 10 units higher than the control mean (20+10) and block 5 10 unites lower.... Make sure you get this logic!

The take home message here is that these numbers make complete sense with respect to what we simulated, and controlling for the among block variation gave us more power to detect a treatment effect, something we would have missed had we not estimated the block source of variation.

6.2.3 Correct Standard Errors for a Figure

When we made our initial plot above, we calculated the standard error based on all observations among blocks. However, the variation we really wish to represent is the variation after having controlled for the blocking effects.

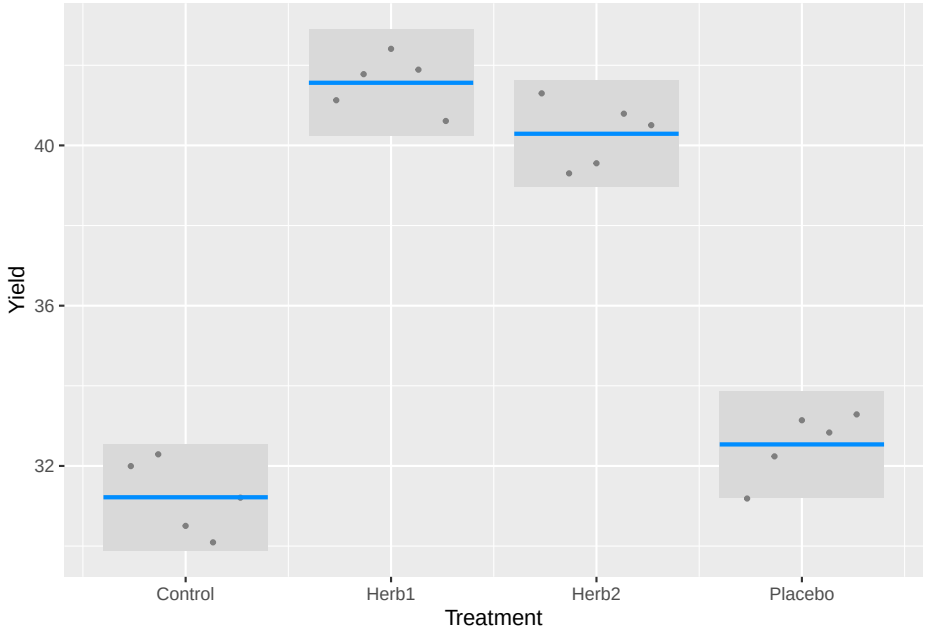
This means that the standard deviation we should probably use is of the error variance from the model: 0.88.

The standard deviation is the $\sqrt{Var}$ and thus, our correct standard errors from the model are $\sqrt{0.88}$

There is a very nice plotting function in the package *visreg* that delivers these proper standard errors in a nice ggplot framework. It presents points are partial

residuals (deviation from the mean for each replicate), lines depciting the means, and shaded area as a 95% confidence interval, calculated 1.96*SE, where the SE is estimated from the model error variance (just above). Compare this to your first graph.

```
visreg(block_model, "treat", gg=TRUE)+
  ylab("Yield") +
  xlab("Treatment")
```



6.2.4 Making inference: confidence intervals and contrasts

We are now in a very strong position to make inference.

Let's start with a rule of thumb linked to the 95% confidence interval. If the CIs don't overlap, they are different; if they do, they are not. This indicates that Cont and Placebo are not significantly different (95% confidence intervals overlap). Herb1 and Herb 2 are significantly different from these, but not each other.

This is OK. But it's not robust. Let's revisit our *post-hoc* and *a priori* methods for evaluating differnces among treatments. We can apply a tukey test and calculate all pairwise differences. This is not a good idea, but let's do it.

```
# use agricolae HSD.test()

tukey_out <- HSD.test(block_model, "treat", group = TRUE)
tukey_out$groups
```

```
##              obs groups
## Herb1   30.46167      a
## Herb2   29.18874      a
## Placebo 21.43581      b
## Control 20.11665      b
```

This confirms our intuition and 95% Confidence Interval insights. But is it correct?

Let's make a formal test of one of the pairwise tests that looks obvious - between Herb1 and Herb2

```
# fit.contrast from gmodels package
contrast <- c(0,-1,1,0)
fit.contrast(block_model, "treat", contrast)
```

```
##                       Estimate Std. Error   t value   Pr(>|t|)
## treat c=( 0 -1 1 0 ) -1.272934  0.6126423 -2.077777 0.05985811
## attr(,"class")
## [1] "fit_contrast"
```

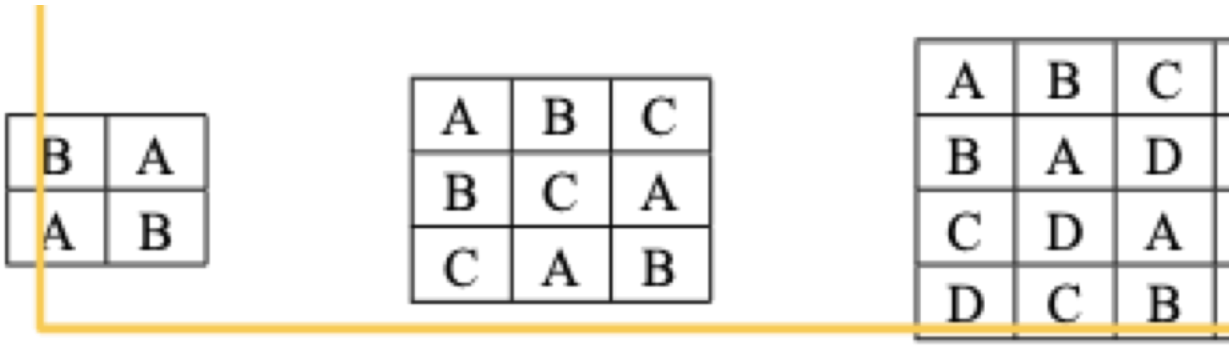
Amazing. The contrast defining a specific test provides a different answer than the post-hoc Tukey test and our guess based on the 95% CIs. Why is that? Which is right?

Of course the contrast is the correct and most reliable result. While both *fit.contrast* and *HSD.test* both manage the model complexity and variance estimates properly, only the contrast reduces the probability of finding a significant difference by chance or failing to find one.

Chapter 7

The Latin Square Design (LSD)

Sometimes you will have two sources of variation in space or time, or perhaps space and time. Perhaps there is a gradient of elevation up and down a hillside, and a gradient of sun incidence orthoganol (90 $^\circ$) to that. There are innumerable examples one could imagine. The principle is that in an LSD, the treatments are allocated randomly in 'rows' and 'columns'.



A Latin Square blocks on both rows and columns simultaneously.

Agricolae has a built in function for helping generate these designs, obvio. Again, new section or new script.....

```
# makes sure random allocation happens the same for everyone.
set.seed(123)

#
```

```
treat<- c("Cont","Herb1","Herb2","Placebo")

# design.lsd
design <- design.lsd(treat,serie = 0)$book
design
```

```
##    plots row col   treat
## 1     11   1   1    Cont
## 2     12   1   2   Herb1
## 3     13   1   3   Herb2
## 4     14   1   4 Placebo
## 5     21   2   1   Herb1
## 6     22   2   2   Herb2
## 7     23   2   3 Placebo
## 8     24   2   4    Cont
## 9     31   3   1 Placebo
## 10    32   3   2    Cont
## 11    33   3   3   Herb1
## 12    34   3   4   Herb2
## 13    41   4   1   Herb2
## 14    42   4   2 Placebo
## 15    43   4   3    Cont
## 16    44   4   4   Herb1
```

Make sure you understand that the double-blocking has happened! See the figure above with 4 letters. Can you create on a piece of paper the matrix from this design you've made in R?

Now, lets generate some data, as before.

```
# for fun, set again.
set.seed(123)

Total.units <- length(treat) * length(treat)
error <- rnorm(Total.units,0,1)

# make sure you think about where we generated variation in the blocking!
design$obs <- 20 +
  (design$treat=="Herb1") * 10 +
  (design$treat == "Herb2") * 9 +
  (design$row==1) * 10 - (design$col==4) * 10 +
  error

# look at it....  there are three explanatory variables...
```

```
# treat, row-block, column-block!
head(design)
```

```
##   plots row col   treat      obs
## 1    11   1   1    Cont 30.18707
## 2    12   1   2   Herb1 41.99603
## 3    13   1   3   Herb2 41.29937
## 4    14   1   4 Placebo 21.12746
## 5    21   2   1   Herb1 31.08111
## 6    22   2   2   Herb2 29.56917
```

7.0.1 Fitting the model

Let's fit the naive and full models as before

```
naive_lsd <- lm(obs ~ treat, data = design)
full_lsd <- lm(obs ~ row + col + treat, data = design)
anova(naive_lsd)
```

```
## Analysis of Variance Table
##
## Response: obs
##           Df Sum Sq Mean Sq F value Pr(>F)
## treat      3 398.31 132.771  2.2902 0.1303
## Residuals 12 695.68  57.973
```

```
anova(full_lsd)
```

```
## Analysis of Variance Table
##
## Response: obs
##           Df Sum Sq Mean Sq F value    Pr(>F)
## row        3 402.44 134.145 139.002 6.209e-06 ***
## col        3 287.45  95.818  99.287 1.672e-05 ***
## treat      3 398.31 132.771 137.578 6.401e-06 ***
## Residuals  6   5.79   0.965
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Very quickly we can see, again, that without blocking, we have a risk of not detecting differences among treatments!

Take a look above at the `design$obs` object. Can you review the `summary` output and validate that the model is estimating what we simulated?

```
summary(full_lsd)
```

```
##
## Call:
## lm(formula = obs ~ row + col + treat, data = design)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.99849 -0.41205  0.00493  0.37463  1.24126
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   30.7965     0.7766  39.654 1.72e-08 ***
## row2         -11.2092     0.6946 -16.137 3.60e-06 ***
## row3         -12.1132     0.6946 -17.438 2.28e-06 ***
## row4         -11.3415     0.6946 -16.327 3.36e-06 ***
## col2           1.2130     0.6946   1.746    0.131
## col3           0.4538     0.6946   0.653    0.538
## col4          -9.1817     0.6946 -13.218 1.16e-05 ***
## treatHerb1    10.2526     0.6946  14.759 6.08e-06 ***
## treatHerb2     9.1602     0.6946  13.187 1.17e-05 ***
## treatPlacebo  -0.4739     0.6946  -0.682    0.521
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9824 on 6 degrees of freedom
## Multiple R-squared:  0.9947, Adjusted R-squared:  0.9868
## F-statistic: 125.3 on 9 and 6 DF,  p-value: 3.921e-06
```

Finally, we can use contrasts to test a few of our key hypotheses. Make some notes on what these things mean!

```
#first the set of comparisons with control
contrast1 <- rbind(
  "C v H1" = c(-1,1,0,0),
  "C v H2" = c(-1,0,1,0),
  "C v P" = c(-1,0,0,1))

fit.contrast(full_lsd, "treat", contrast1)
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## treatC v H1 10.2525618  0.6946426 14.7594771 6.079845e-06
## treatC v H2  9.1601564  0.6946426 13.1868620 1.174267e-05
## treatC v P  -0.4738879  0.6946426 -0.6822038 5.205624e-01
```

```
## attr(,"class")
## [1] "fit_contrast"
```

```
# now the H1 vs H2
contrast2 <- rbind("H1 v H2" = c(0,-1,1,0))
fit.contrast(full_lsd, "treat", contrast2)
```

```
##               Estimate Std. Error   t value  Pr(>|t|)
## treatH1 v H2 -1.092405  0.6946426 -1.572615 0.1668691
## attr(,"class")
## [1] "fit_contrast"
```

Chapter 8

Designs for testing for interactions: factorial designs.

8.1 Introducing Interactions

Before jumping into an example, let's introduce a simple statement that forms the core of both asking and interpreting *interactions*.

If there is an interaction between two explanatory variables, X and Z, on our response variable Y, then:

> The effect of X on Y varies by Z. **OR** The effect of X on Y depends on Z.

This use of *varies by* **OR** *depends on* defines context dependency and that's what defines interactions.

Now, let's return to our CO2 and UV-B solar radiation experiment in the Arctic Tundra example to introduce the idea of an interaction again.

Context: an arctic tundra study Increasing ultraviolet-B (+UV-B) radiation from ozone depletion (the arctic ozone hole) Increasing atmospheric CO2 (+CO2) from anthropogenic emissions For plants: UV-B potentially harmful, +CO2 potentially beneficial Therefore +CO2 could alleviate UV-B damage impacts.

Hypotheses +UV-B radiation will reduce the growth of arctic plants +CO2 will increase the growth of arctic plants +UV-B radiation impacts will be less under +CO2

What is unique about these context and hypotheses? It's the presentation of CO2 and UVB in the same statement and the use of words like "the effect of X will alleviate the impacts of Y" and words like "the effects of X will be less under Y". These words and phrases reflect the context dependency of the effects of treatment levels.

Thus, to restate what we introduced above…. when we talk about interactions, we can rely on a very simple vocabulary that is independent of the actual treatment levels: we can always describe an interaction like this:

> The effect of X on Y depends on Z.

or

> The effect of X on Y varies by Z.

In this 'rubric', Y is the response variable, and X and Z are explanatory, independent variables. So, in our example above,

> the effect of CO2 on plant biomass yield depends on UV-B radiation levels.

OR

> the effect of CO2 on plant biomass yield varies with UV-B radiation levels.

This simple phrasing describes any interaction.

Here are some numerical examples to drive home the point.

Control = 20g Yield UV-B = 10g Yield C02 = 29g Yield

ADDITIVE RESULT: C02 + UV-B = 39g Yield SYNERGISTIC RESULT: C02 + UV-B = 60g Yield ANTAGONISTIC RESULT: C02 + UV-B = 19g Yield

8.1.1 Let's design an experiment, again…

Your resources:

The arctic research station has given permission for 16 plots (each 2m x 2m) in the natural vegetation nearby. - One UV-B plot (2m x 2m) costs £4000 (this provides the UV-B lamps, frame, power and control system, wooden walkways around the plots) - One CO2 plot (2m x 2m) costs £6000 (this provides a CO2 release system, CO2 control and covers CO2 purchase costs, wooden walkways

around the plots) - One control plot (2m x 2m) costs £200 (marking posts, wooden walkways around the plots) - Yes, the CO2 supply system does fit underneath the UV-B lamps.

You have a budget of £86,000

Design an experiment to test the hypotheses. (An answer is on the BB Site)

8.1.2 The Factorial Design: How and Why Study Interactions?

The UV-B and C02 experiment could be thought of as two experiments a Control vs UV-B and a Control vs C02 experiment. If we combine these we get a Factorial Experiment where the treatments are

- Control
- UV-B
- C02
- UV-B + C02

So we have all combinations of treatments.

But do we treat each of these as unique treatments (e.g. the one-way ANOVA example)? No, we don't. We design and analyse the data we collect in a two-way analysis - a factorial design. Two-way ANOVA (and ANCOVA - see later), are the toolboxes.

Factorial treatments have two main advantages. When factors *DO interact* – so the effect of C02 depends on UV-B – then we can estimate the interaction - the dependency. *One-way designs* cannot do this, and can lead to serious misunderstandings (because we are assuming that the effect of one thing DOES NOT depend on the other).

Furthermore, when factors *DON'T interact*, *factorial designs* are more efficient (smaller error variance) than one-way designs experiments.

Hence ALWAYS use factorial designs when you can!

8.2 A Factorial Design and the Two-Way ANOVA

I'd recommend a new script now for this section. It's a pretty distinct example. Don't forget to use library() at the top to get all those packages working for this script. And don't forget, this script can live in the same R Project as the other one(s).

```
set.seed(123)

Nreps <- 4 # 4 replicates per treatment level
trt <- c(2,2) # sets out the 2-way design
design <- design.ab(trt,r=Nreps,serie=0,design="crd")$book
head(design)
```

```
##   plots r A B
## 1     1 1 2 1
## 2     2 1 1 2
## 3     3 1 1 1
## 4     4 1 2 2
## 5     5 2 2 2
## 6     6 2 2 1
```

So, the `design.ab` function generates a nice picture, but uses numbers instead of words for the various treatments and has set Letters as the treatments.

We don't support this approach - you WILL forget what the numbers and letters mean, so let's change these CODES to represent CO2 and UVB REALITY. We'll use `fct_recode` from the tidyverse package called `forcats` (for categories, but we like cats).

```
design <- design %>%
  rename(UVB = A, CO2 = B) %>%
  mutate(UVB = fct_recode(UVB, "Con" = "1", "UVB+" = "2"),
         CO2 = fct_recode(CO2, "Con" = "1","CO2+" = "2"))

head(design, 10) # is this what you expect
```

```
##    plots r  UVB  CO2
## 1      1 1 UVB+  Con
## 2      2 1  Con CO2+
## 3      3 1  Con  Con
## 4      4 1 UVB+ CO2+
## 5      5 2 UVB+ CO2+
## 6      6 2 UVB+  Con
## 7      7 3 UVB+ CO2+
## 8      8 2  Con  Con
## 9      9 3  Con  Con
## 10    10 4  Con  Con
```

NOTE 1: this is using the `fct_recode` function from the *forcats* package in the *tidyverse* collection.

NOTE 2: we used the design = "crd" argument to specify a completely randomised, 2-way factorial design... you could make this a block design with the "rcbd".... see the help file for `design.ab`

8.2.1 Build the Data

Here we will build two sources of data - one that has NO INTERACTION, and one that DOES.

```
#Sample Sizes and errors
Total.units <- 2 * 2 * Nreps
error <- rnorm(Total.units,0,3)

# the data with no interaction
design$obs <- 20 -
  (design$"UVB"=="UVB+") * 10 +
  (design$"CO2" == "CO2+") * 9 +
  error

# the data with interaction
design$obs2 <- 20 -
  (design$"UVB"=="UVB+") * 10 +
  (design$"CO2" == "CO2+") * 9 +
  # when both are together add 8 more (synergy)
  ((design$"UVB"=="UVB+") & (design$"CO2" == "CO2+")) * 8 +
  error

head(design)
```

```
##   plots r  UVB  CO2      obs     obs2
## 1     1 1 UVB+  Con 12.82805 12.82805
## 2     2 1  Con CO2+ 25.38908 25.38908
## 3     3 1  Con  Con 19.17974 19.17974
## 4     4 1 UVB+ CO2+ 25.85980 33.85980
## 5     5 2 UVB+ CO2+ 15.92984 23.92984
## 6     6 2 UVB+  Con  7.75704  7.75704
```

8.2.2 Plot the data!

Here we combine some dplyr magic (calcuating means in each group - there are two grouping variables!), some ggplot magic (adding the lines connecting the means on top of the raw data) and the beauty of patchwork, the package for plot layouts. Don't forget to check that you've got all of the `library()` calls at the top of your script, and that you ran them!

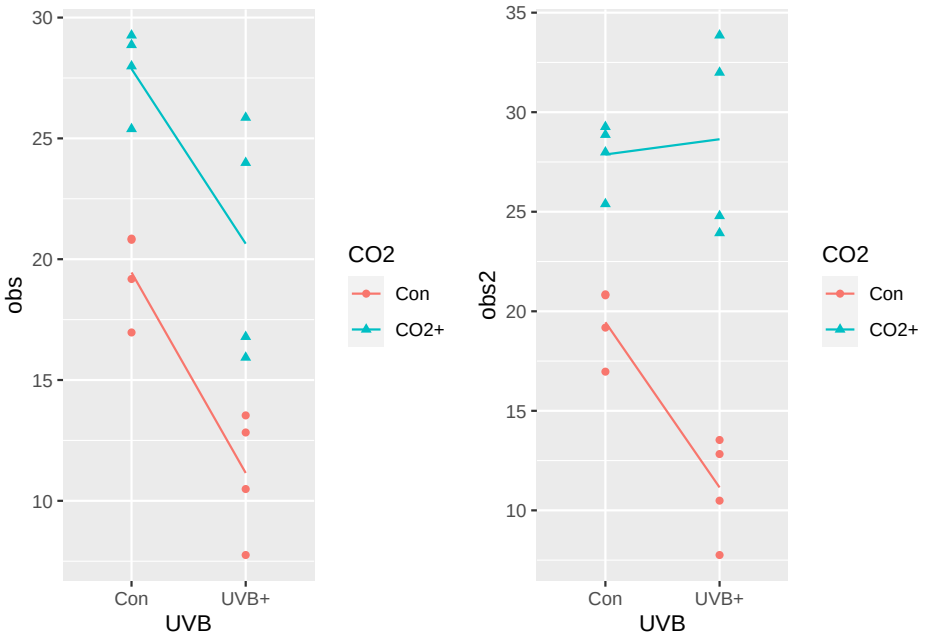
```
# No interaction summary
sumDat1 <- design %>%
  group_by(UVB, CO2) %>%
  summarise(
    obs = mean(obs)
  )

# with interaction summary
sumDat2 <- design %>%
  group_by(UVB, CO2) %>%
  summarise(
    obs2 = mean(obs2)
  )

# no interaction
p1 <- ggplot(design, aes(x = UVB, y = obs, colour = CO2, shape = CO2, group = CO2))+
  geom_point()+
  geom_line(data = sumDat1)

# with interaction
p2 <- ggplot(design, aes(x = UVB, y = obs2, colour = CO2, shape = CO2, group = CO2))+
  geom_point()+
  geom_line(data = sumDat2)

p1+p2
```



What you see above on the left, is a pattern that suggest that the effect of _____ on _____ does not vary by _____. In contrast, on the right.....:

8.2.3 Analyse the data

```
# model with no interaction
int_mod_1 <- lm(obs ~ CO2*UVB, data = design)

# model with interaction
int_mod_2 <- lm(obs2 ~ CO2*UVB, data = design)

# model with interaction, but specified incorrectly.
int_mod_3 <- lm(obs2 ~ CO2 + UVB, data = design)

anova(int_mod_1)
```

```
## Analysis of Variance Table
##
## Response: obs
##           Df Sum Sq Mean Sq F value    Pr(>F)
## CO2        1 321.00  321.00  33.469 8.671e-05 ***
## UVB        1 241.27  241.27  25.156 0.0003013 ***
```

```
## CO2:UVB    1   1.13    1.13   0.118 0.7371854
## Residuals 12 115.09    9.59
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(int_mod_2)
```

```
## Analysis of Variance Table
##
## Response: obs2
##           Df Sum Sq Mean Sq F value    Pr(>F)
## CO2        1 671.67  671.67 70.0311 2.364e-06 ***
## UVB        1  56.75   56.75  5.9165   0.03159 *
## CO2:UVB    1  82.15   82.15  8.5654   0.01268 *
## Residuals 12 115.09    9.59
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(int_mod_3)
```

```
## Analysis of Variance Table
##
## Response: obs2
##           Df Sum Sq Mean Sq F value   Pr(>F)
## CO2        1 671.67  671.67  44.269 1.58e-05 ***
## UVB        1  56.75   56.75   3.740  0.07519 .
## Residuals 13 197.24   15.17
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note the differences in the outputs. Not what we infer if we model the interaction data incorrectly. There is no free lunch. You must understand your data and your question!

8.2.4 final figuring

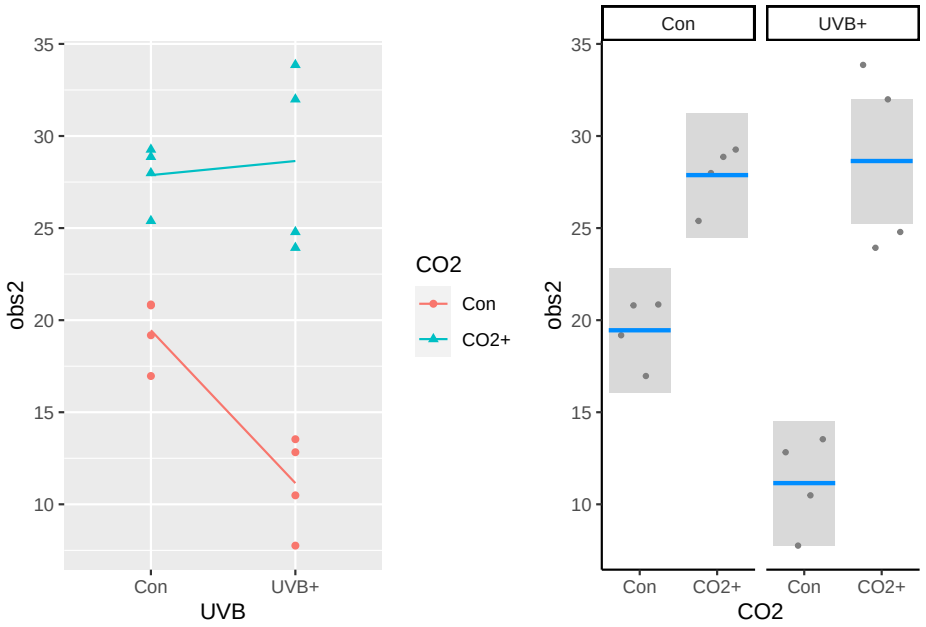
Don't forget that the correct standard error for the 'result' is the residuals mean squared. You can use the dplyr+ggplot2 method, or visreg.

Here we use visreg and put the figure next to our original ggplot for the interaction data.

We plot the correct model for the interaction data.

```
p3 <- visreg(int_mod_2,"CO2",by="UVB",gg=TRUE) +
  theme_classic()

p2+p3
```



Chapter 9

Experimental Design and ANOVA - A Summary

There are three sets of things you should now know about based on reading these chapters and working through the examples. The first set is a bunch of clever R stuff for helping generate experimental designs. The second set is a better understanding of experimental designs in general. The third set is how to analyse and interpret data from 1-way and 2-way ANOVA models, that are the toolboxes for analysing these designs.

Lets review these in reverse order

9.1 How To Analyse ANOVA models

In semester 1, you were introduced to the 1-way ANOVA model, using the daphnia parasite data. Here you've learned more about those analyses.

You now understand better what treatment contrasts are, the risk of post-hoc tests like the Tukey Test, and methods for making *a priori* contrasts to test specific hypothesis. You learned how specify specific hypoptheses and use the `fit.contrast` function from the *gmodels* package to help you be really precise in asking and answering questions.

You also learned how to include a blocking factor in your models to control for environmental gradients in your experiments. You learned how to estimate the standard error that is used for the testing of among treatment level differences too.

Finally, you learned about the idea of interactions - the effect of UVB on Plant Yield Varies by CO2 level. You learned how to fit these models (e.g. `lm(Yield ~`

`UVB * CO2, data = dataframe`)) and how to interpret the statistical outputs against the graphs you made. Speaking of which, you learned how tomake graphs that reveal the presence of interactions, or not.

9.2 Experimental Designs.

You should now understand the differences between, and value of the Completely Randomised Design, the Randomised Complete Block Design and the Latin Square Design. There are plenty more designs out there to manage lots of different issues. The *agricolae* package is a great tool for generating designs (to help plan experminents) and is also a great resource for learning about them.

https://myaseen208.github.io/agricolae/articles/Intro_agricolae.html

9.3 Clever R Stuff

You've also increased your R ninja skills. You've learned about building 'fake data' using the `design_` class of functions in *agricolae* mixed with random numbers and a clear set of tools that develop understanding of how additive and interactive effects arise among treatments in your data. This is a super skill.

There is nothing better than making data that has a pattern or not, to understand better where patterns are in your own data. If you can build 'fake data' with and without the pattern you might expect from theory, then you have the template on which to examine carefully your 'real data' collected via the experiments or surveys.

Chapter 10

Introducing the ANCOVA - the analysis of covariance

In the previous chapters, we introduced classic experimental design principles, and in doing so, we focused on revisiting the 1-way ANOVA and introduced the 2-way ANOVA. In these models, the explanatory variables are categorical (they are factors). This means explanatory variable has distinct, discrete categories or *levels*.

In many of our experiments, however, we might combine these categorical variables with a continuous variable. For example, we might estimate eggs produced as a function of body size and season or we might estimate the effect on plant growth rates of soil moisture in high versus low Nitrogen (N) conditions.

These simple examples describe an ANCOVA, where the one explanatory variable is continuous (e.g. body size or soil moisture) and the other is categorical (e.g. season or N-level). When this is the case, we are essentially combining the ANOVA with the Regression. If we recall that regression is largely about estimatinge slopes and intercepts, we might think, hey, COOL, so in an ANCOVA, we can ask if the categorical variable alters the slope or not…..

What's very important to understand is that our core statement about interactions does not change.

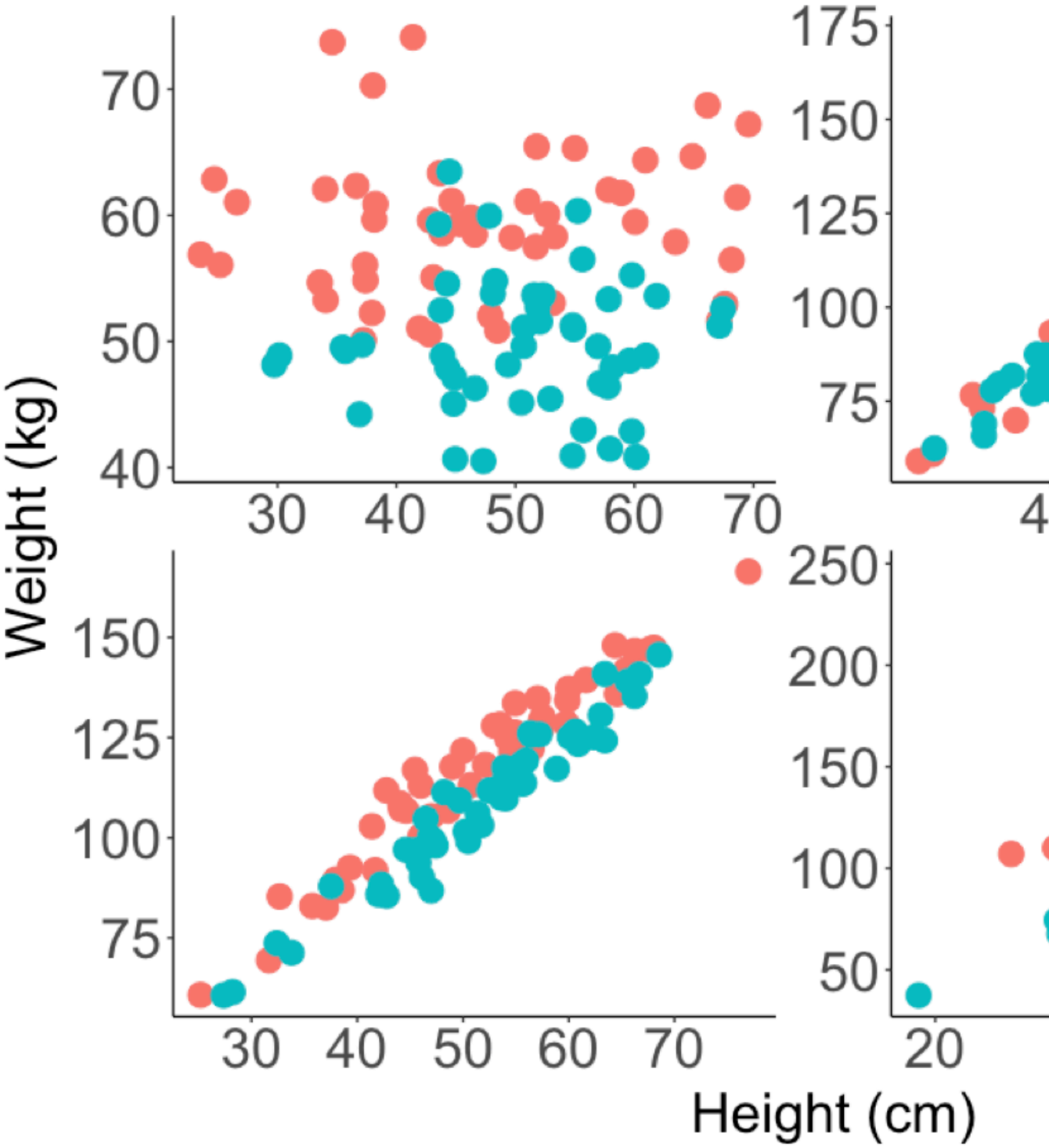
> The effect of X on Y varies by Z, translates to a) The effect of body size on egg production varies by season; or b) the effect of soil moisture on growth rate varies by N-level

When written like this, the previous statements about slopes should be even more clear. The effect of body size (continuous) on egg production is a regression and we estimate a slope. We then can ask whether this slope is different between seasons.

10.1 Setting up the various ideas.

Let's start by looking at an example where the effect of Height on Weight varies by Sex. This is a classic set of data from numerous organisms.... it captures biologically the question about sexual dimorphism - does the relationship between Height and Weight (a regression) vary by Sex (ANOVA)?

This relationship can take on many patterns.


Weight (kg)
Height (cm)
70
60
50
40
30
40
50
60
70
175
150
125
100
75
4
150
125
100
75
30
40
50
60
70
250
200
150
100
50
20

- In the upper left we see a pattern where Males are heavier than Females, but there is no effect of Height on Weight.
- In the upper right, we see that there is a positive relationship between Weight and Height, but no differences between Males and Females.
- In the lower left, we might argue that there is a positive relationship between Weight and Height, that the relationship between Weight and Height does not vary (the slope is the same) and that Males are heavier (the red dots are mostly above the blue)
- In the lower right, we see evidence of an interaction - the effect of Height on Weight (the slopes), varies by Sex.

It's quite important to recognise that each of patterns is possible outcome to testing the SAMEa hypothesis. The key thing to remember is that we have an a priori (in advance hypothesis). Regardless of the pattern, we should specify the appropriate model to test your the hypothesis and answer the question that motivated the design of the experiment and data collection.

For example, in these data, we fundamentally start with the question - does the effect of Height on Weight vary by Sex. The data might look like any of the above patterns. But as we started with that, we have to try and answer that question. And there is only one model syntax in the above figure that does this: `Weight ~ Height * Sex`. We'll expand on what this means as we work through the example.

10.2 Working through and example.

Let's work with a built in dataset in R - the Davis Study, which is exactly these data. The associated assignment is another example.

The process of doing this will follow a tried and true approach to analysing data. You should embed this workflow in your head:

1) PLOT the Data
2) Build the model to test the hypothesis
3) Check the Assumptions
4) Make Inference
5) Update the Figure for Publication

10.2.1 Get the data and make your picture

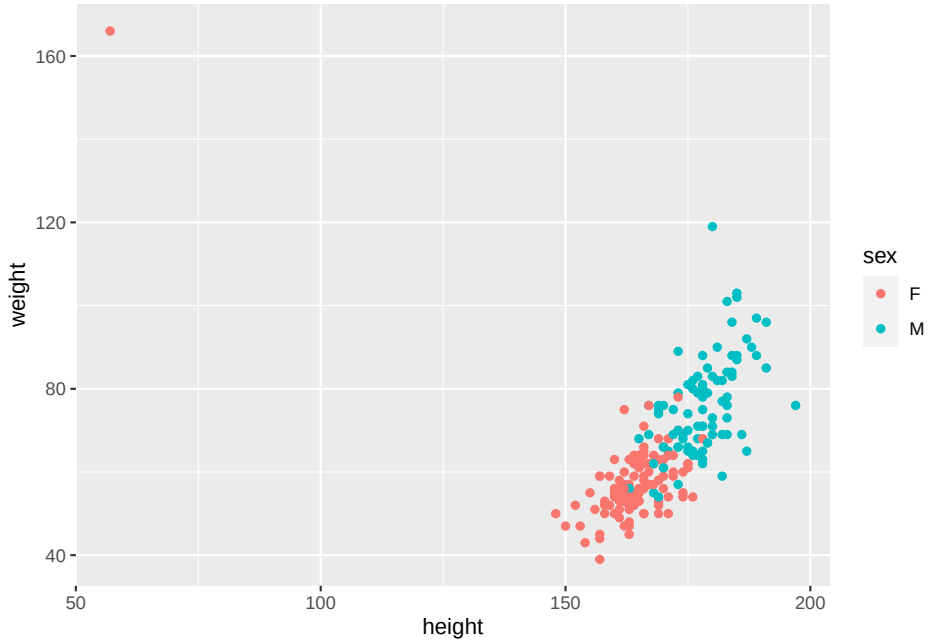
The data are built into R, but embedded in the `carData` package, which was installed with the `car` package.

```
Davis <- carData::Davis

glimpse(Davis)
```

```
## Rows: 200
## Columns: 5
## $ sex    <fct> M, F, F, M, F, M, M, M, M, M, M, F, F, F, F, F, M, F, M, F, ...
## $ weight <int> 77, 58, 53, 68, 59, 76, 76, 69, 71, 65, 70, 166, 51, 64, 52,...
## $ height <int> 182, 161, 161, 177, 157, 170, 167, 186, 178, 171, 175, 57, 1...
## $ repwt  <int> 77, 51, 54, 70, 59, 76, 77, 73, 71, 64, 75, 56, 52, 64, 57, ...
## $ repht  <int> 180, 159, 158, 175, 155, 165, 165, 180, 175, 170, 174, 163, ...
```

```
ggplot(Davis, aes(x = height, y = weight, col = sex))+
  geom_point()
```



Ahh…. first hurdle…. what's going on here? Well, always plot your data is totally justified, because it looks like one of the data points has the Height and Weight data entered incorrectly.

Let's fix that. Lets find the row, and make the change

```
#which row? 12
Davis %>% filter(height < 100)
```

```
##    sex weight height repwt repht
## 12   F    166     57    56   163
```

```
# BASE R syntax to see each observation
Davis$weight[12]
```

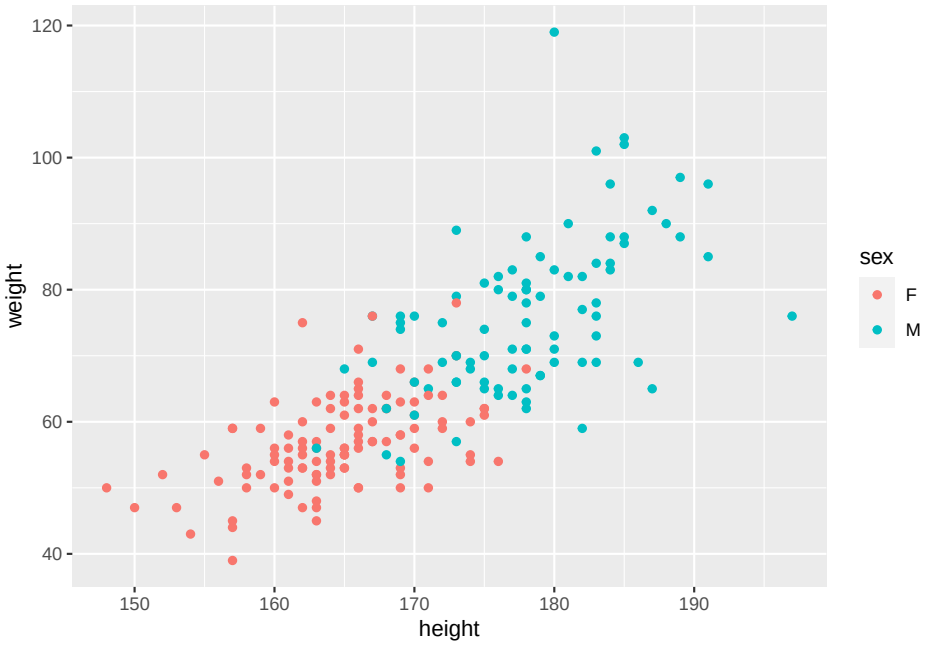
```
## [1] 166
```

```
Davis$height[12]
```

```
## [1] 57
```

```
# BASE R syntax to change them
Davis$weight[12] <- 57
Davis$height[12] <- 166

# replot
ggplot(Davis, aes(x = height, y = weight, col = sex))+
  geom_point()
```

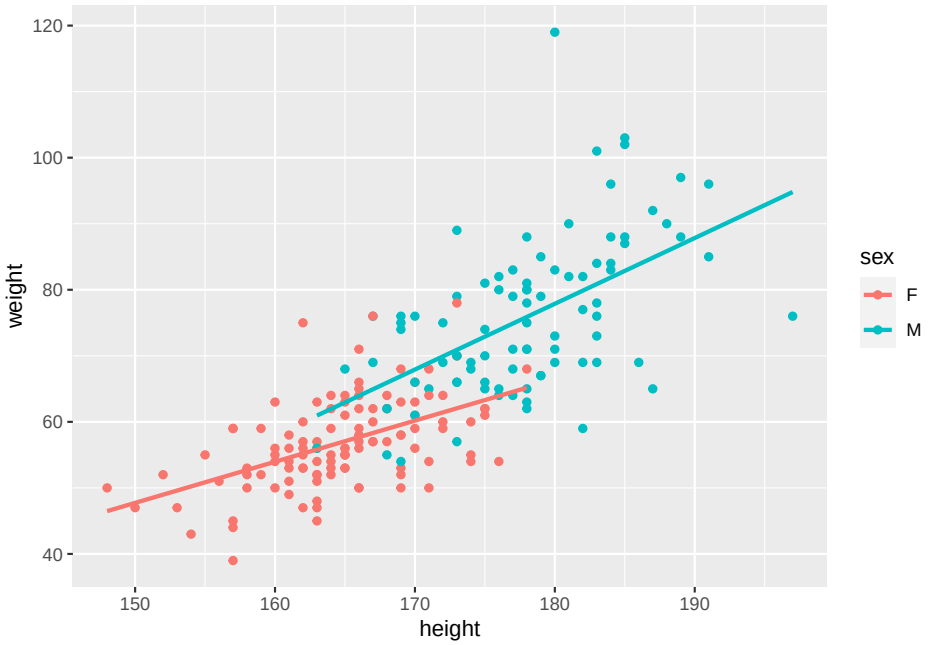


Excellent. Lets look at this picture and ask ourselves whether the effect of Height on Weight appears to vary by Sex? What do you think?

We can actually get a bit of help here. We can use some ggplot magic to help us make this GUESS. > NOTE: this is not doing statistics. This is using graphics to help guide our expectation of the outcome of doing the statistical test of our hypothesis.

```
ggplot(Davis, aes(x = height, y = weight, col = sex))+
  geom_point()+
  # add a best fit line to each group (the sex category)
  geom_smooth(method = lm, se = FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Again, we have not 'proven' anything or tested our hypothesis. What we have is a good guess though. We can guess that

1. Males are heavier than females (what is the difference in the intercepts?)
2. The effect of height on weight is positive (the slope(s) are positive, not negative)
3. There might be a difference in slopes - the Male line is steeper.

We might even go as far to estimate by eye the slope assuming that there is no effect of Sex. Recalling that the slope is the rise/run or the change in y over the change in x, we can guess that the slope is ~ (100-40)/(200-140) = (60/60) = 1. We are not expecting much variation around that. But if we see the statistics returning a value 10x more or less than this, we should be concerned!

10.3 Building the model (and understanding it)

The next step is to build the model to test our hypothesis. As we declared above, the model to test the interaction is `lm(weight ~ height * sex, data = Davis)`. Let's discuss this a bit.

First, this model expands to the following full description:

```
lm(weight ~ height + sex + height:sex, data = Davis)
```

This reads as "Weight is a function of the effect of Height (slope), Sex (intercept) and the interaction between Height and Sex (do the slopes vary?). The `Height * Sex` syntax always expands to this *full model* - the model containing the main effects and the interaction.

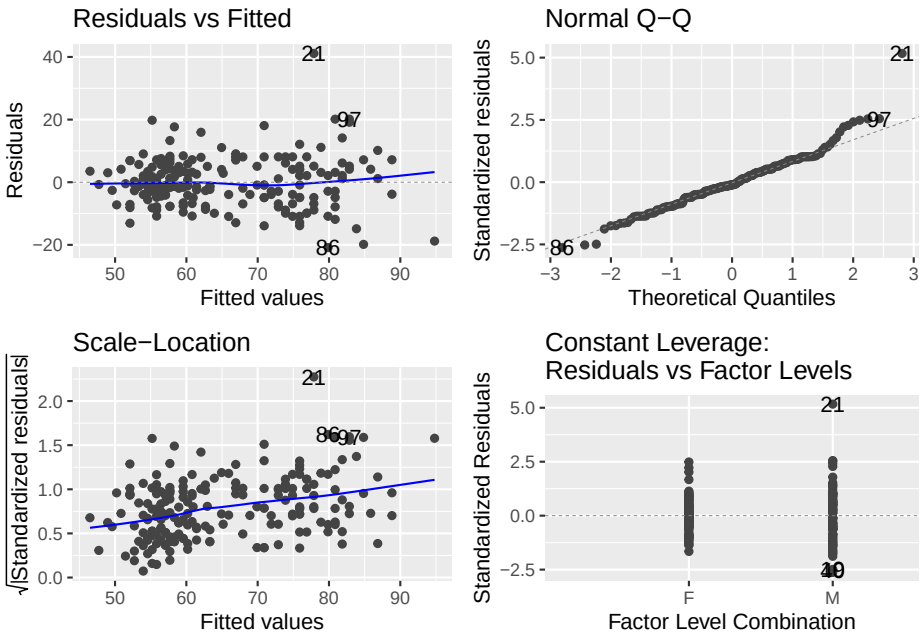
OK. Let's fit the model

```
# we call the model mod_Davis.
mod_Davis <- lm(weight ~ height + sex + height:sex, data = Davis)
```

That's it. We've got the model. But before we make any attempts at actually evaluating our test of the hypothesis, we have to check the assumptions!

To do this, we use the autplot function from the ggfortify package.

```
autoplot(mod_Davis)
```



OK. Let's walk through the three core diagnostics.

In the upper left, we are evaluating the systematic part of the model - are there any systematic departures from the linear model we've speficied? Are there interactions missing or specific nonlinearities? Nope.

In the upper right, we are evaluating the normality of the residuals. These too look pretty good.

In the lower left, we are evalauting the assumption of a constant mean-variance relationship. Oops.

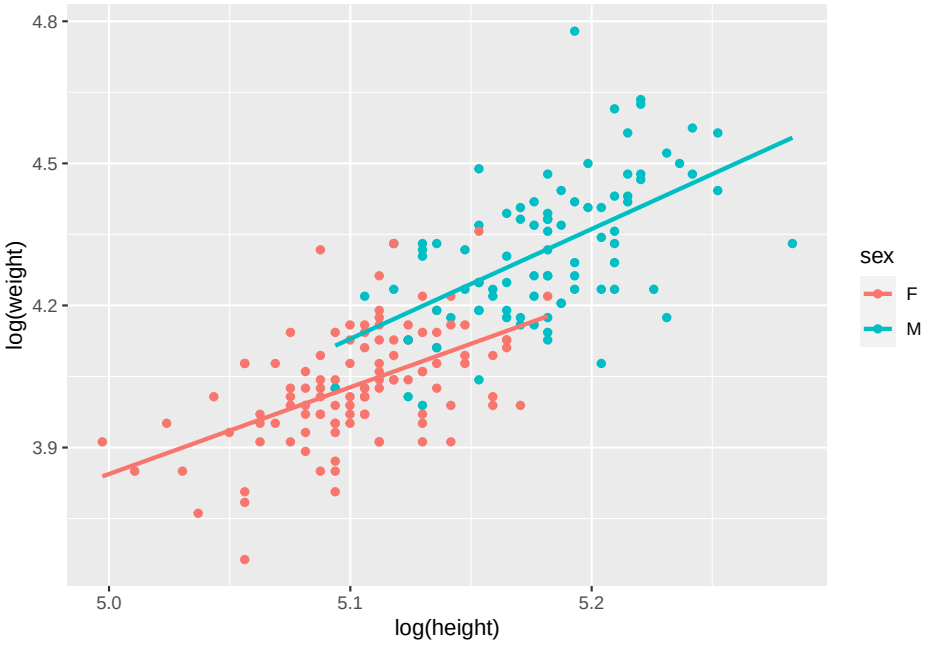
10.3.1 Dealing with some issues with residuals.

The above issue is one we can deal with via a *transformation*. Let's start with the facts…. When the variance increases with the mean, one transformation that can work is the logarithm of the continuous variables.

ADD DYLANS STUFF FROM MEE PAPERS and Mean - VARIANCE

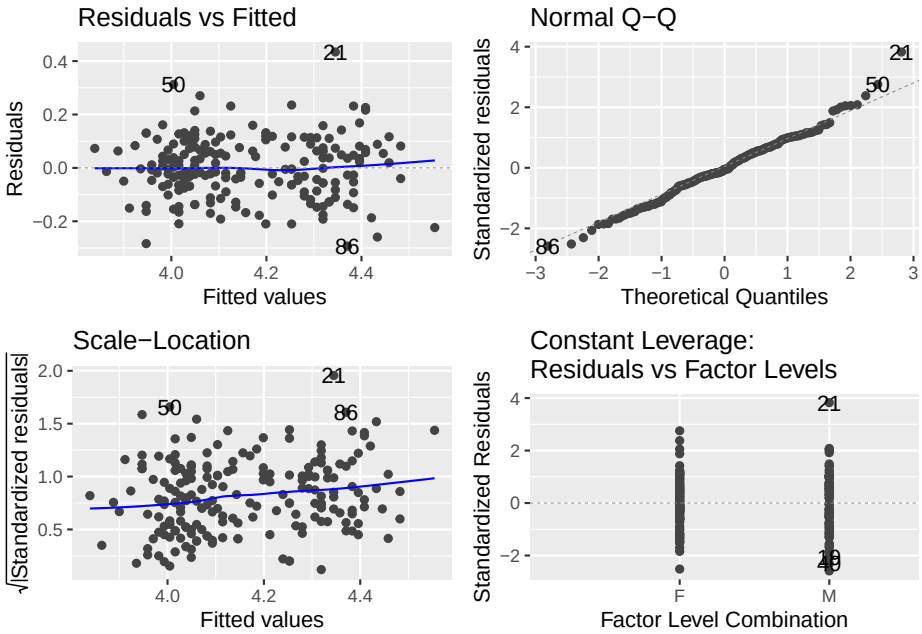
```
# re-plot with log-log
ggplot(data = Davis, aes(x = log(height), y = log(weight), col = sex))+
  geom_point()+
  geom_smooth(method = lm, se = FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# fit the model - note we can specify the transformation RIGHT IN THE MODEL
mod_Davis_log <- lm(log(weight) ~ log(height) * sex, data = Davis)

# check the new residuals
autoplot(mod_Davis_log)
```



This is definitely flatter - you can tell by the range on the y-axis of the scale-location plot.

So, we can now move to evaluating the model. You are probably thinking about what it means to analyse the data on the log-log axis…. we will come to that shortly.

10.3.2 Making inference on the ANCOVA

OK. So, the next step is to use review the ANOVA table - yes, we use the same Anova function to build an anova table to explore the results of an ANCOVA.

Let's revisit our initial guesses and work through what the model is telling us:

1. Males are heavier than females (what is the difference in the intercepts?)
2. The effect of height on weight is positive (the slope(s) are positive, not negative)
3. There might be a difference in slopes - the Male line is steeper. ‘

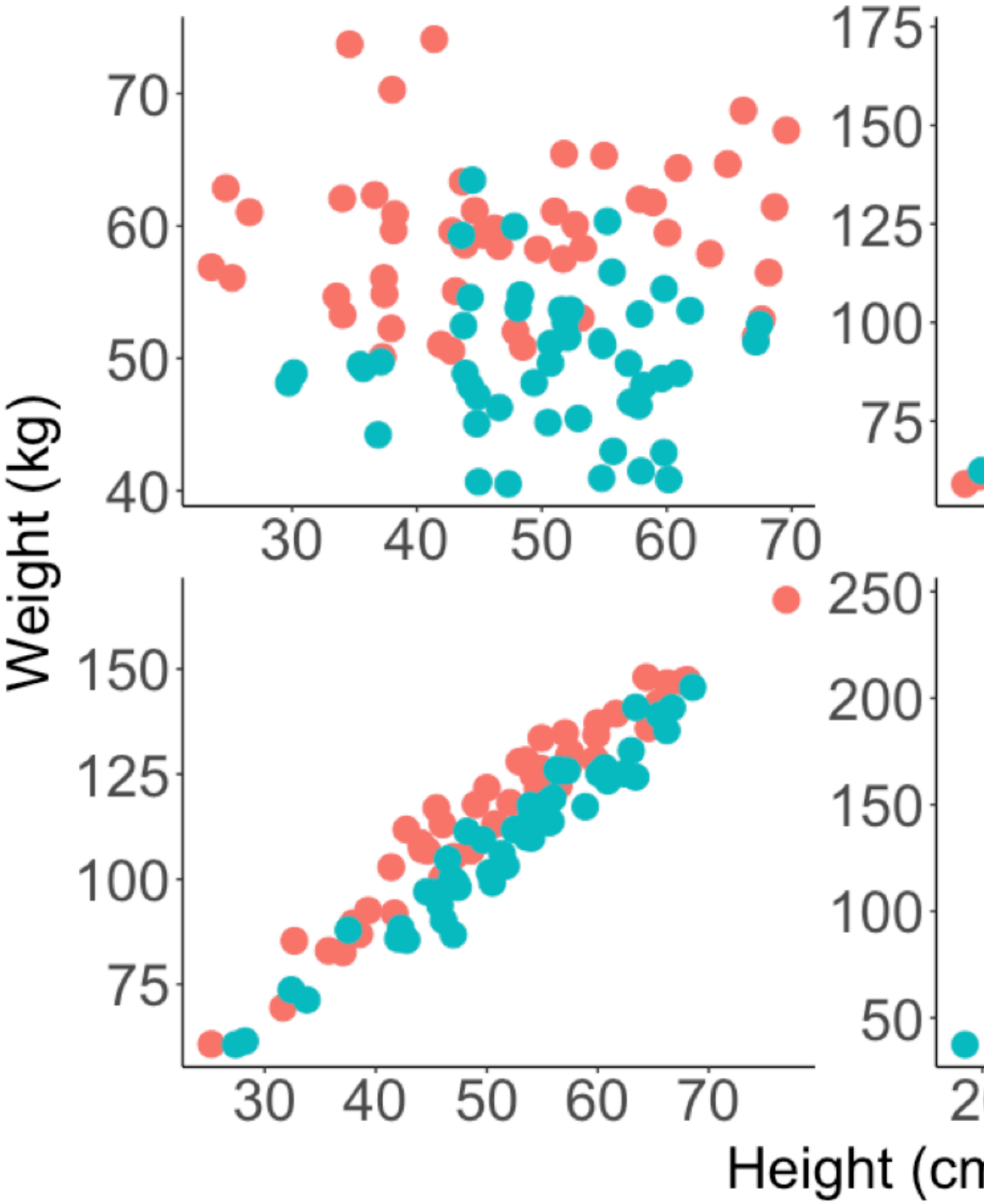
```
anova(mod_Davis_log)
```

```
## Analysis of Variance Table
##
## Response: log(weight)
##                  Df Sum Sq Mean Sq  F value    Pr(>F)
## log(height)       1 4.6643  4.6643 357.4735 < 2.2e-16 ***
## sex               1 0.3446  0.3446  26.4115 6.647e-07 ***
## log(height):sex   1 0.0144  0.0144   1.1038    0.2947
## Residuals       196 2.5574  0.0130
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The table tells us that there is no evidence for an interaction. The last line of this table contains the p-value for the interaction. It is testing the null hypothesis that the slopes are the same. We can not reject this null hypothesis. There is no evidence that allowing the slopes for Males and Females be different is supported. That's the answer to guess 3 above. It is also the answer to our question: does the effect of Height on Weight vary by Sex. Nope.

However, we do detect *main effects* of height and sex. This means that each of these DOES have an effect - but we say the effects are additive.

Great. What this means is that the data actually conform to picture in the lower left.


Weight (kg)
70
60
50
40
30
40
50
60
70
175
150
125
100
75
150
125
100
75
30
40
50
60
70
250
200
150
100
50
Height (cm

There is a slope associated with Height (we don't know whether it's positive or not, but we think it is), and there is an effect of sex (we don't know whether Males are heavier yet). What we can say, with confidence, is this.

> The effect of height on weight did not vary by sex (F = 1.01, df = 1,196, p = 0.29). The effect of sex (F, df, p) and height (F, df, p) are thus additive on Weight.

Another way to read this is the following:

When we allow for a common slope associated with logHeight, we capture 4.66 units of variation in weight, which when compared to the 0.013 units of variation left over (residual MSE), is alot. The F is thus BIG and the p-value small. When we allow for an effect of Sex to define different average weights, this too is important as it captures 0.344 units of variation in weight, which compared to the 0.013 units in the residual, is also big (Big F, small P). However, when we allow for the slopes caused by height to vary with sex, this captures very little variation with respect to the residual variation (0.014 comprared to 0.013) hence a small F and a large P.

10.3.3 The summary table and making sense of the log stuff.

Let's start investigating the actual details via the `summary` table:

```
summary(mod_Davis_log)
```

```
##
## Call:
## lm(formula = log(weight) ~ log(height) * sex, data = Davis)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29310 -0.06543 -0.00592  0.07420  0.43410
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -5.3202     1.6016  -3.322  0.00107 **
## log(height)        1.8329     0.3138   5.841 2.13e-08 ***
## sexM              -2.3724     2.3765  -0.998  0.31936
## log(height):sexM   0.4852     0.4618   1.051  0.29473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1142 on 196 degrees of freedom
## Multiple R-squared:  0.6626, Adjusted R-squared:  0.6575
## F-statistic: 128.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

This table reports on the *treatment contrasts* we introduced earlier this semester. Let's walk through the details.

The first two rows correspond to one of the sexes. Can you identify which one? The hint is looking at the other rows. They have an M after them. M comes after F. So the first two rows correspond to the intercept and slope of the Females.

You may be asking why the intercept is a negative number. A simple look back at the range of the x-axis should answer this… the y intercept occurs at x = 0, right?

You should understand, from the principles of an equation for a line, the following:

> the effect of height on weight for females is `log(weightF) = -5.31 + 1.82*log(heightF)`.

OK… let's interpret what the next two lines mean. The third line is labelled `SexM`. The fourth is labelled `logHeight:SexM`. Any guesses as to which is associated with the slope and which the intercept?

One trick, again, is to recognise the syntax -> the presence of the `:` is indicative of the interaction which is the thing that lets slopes vary. So, `SexM` is about intercepts and `logHeight:SexM` is about slopes. But what are they?

Please recall from previous work that treatment contrasts are about DIFFERENCES. Thus, the `sexM` term is a difference between the female and make intercept and the `logHeight:SexM` is about the difference between the female and make slopes.

This allows the following maths….

`log(weightF) = -5.31 + 1.82*log(heightF)`

`log(weightM) = (-5.31-2.37) + (1.82+0.48)*(log(heightF))`

However, we know that the 0.48 increase in the slope is not siginficant, and as a result, the -2.37 increase in the intercept is also not correct.

10.3.4 The equations supported by the model.

In order to identify the equations supported by the model, we can refit the model to reflect the ADDITIVE effects only.

```
mod_Davis_log2 <- lm(log(weight) ~ log(height) + sex, data = Davis)
summary(mod_Davis_log2)
```

```
##
## Call:
## lm(formula = log(weight) ~ log(height) + sex, data = Davis)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28714 -0.06906 -0.00978  0.07615  0.43717
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.46349    1.17541  -5.499 1.18e-07 ***
## log(height)  2.05688    0.23030   8.931 2.99e-16 ***
## sexM         0.12417    0.02417   5.138 6.66e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1143 on 197 degrees of freedom
## Multiple R-squared:  0.6607, Adjusted R-squared:  0.6573
## F-statistic: 191.8 on 2 and 197 DF,  p-value: < 2.2e-16
```

Right, now we get our expected pattern:

`log(weightF) = -6.46 + 2.05*log(heightF)`

`log(weightM) = (-6.46 + 0.12) + 2.05*log(heightF)`

We can see the common slope and the slightly higher average weight (0.12 units!) of males.

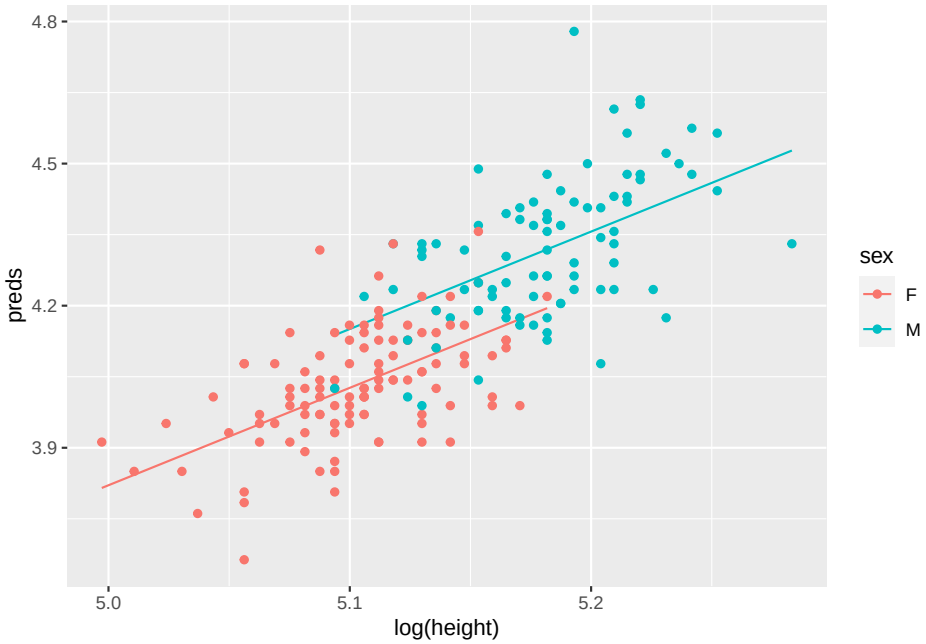
10.3.5 Making even more inference and drawing the picture

OK, lets use some plotting tricks and get a nice picture:

```
# first create a data frame to use that combines the raw data
# with the predictions from the model
# note that the model is the ADDITIVE model, for plotting
plotThis <- Davis %>%
  mutate(preds = predict(mod_Davis_log2))

# now, make lines with the model predictions
# and add the points as the raw data
```

```
# pay attention to where we are using log()!
ggplot(plotThis, aes(x = log(height), y = preds, col = sex))+
  geom_line()+
  geom_point(aes(y = log(weight)))
```



Great. Now we can see that the full model (mod_Davis_log) told us. The lines are parallel. There is no interaction.

In fact, we can do a bit more here.

Remember that the model says

For women log(weight)= -6.46 + 2.05 * log(height) For men log(weight)= (-6.46 + 0.12) + 2.05 * log(height)

Because we are in log land, we can do some clever maths: We know that

`log(weight Female) - log(weight Male) = - 0.12`

(this is the treatment contrast for the intercept). We also know (please memorise) that the difference between logs is equal to the log of the ratio:

`log(weightF/weightM) = -0.12`

Finally, we can convert both sides to non-log by exponentiating:

`exp(log(weightF/weightM)) = exp(-0.12) = 0.88`

What does this mean? At any given height, female weight is 88% of a makes (12% lower).

Cool! That's one of the benefits of log-log linear relationships. The ratio of the categories in log-log land directly translates to a percent difference between the groups.

It's also worth noting that the height slope is ~2, which suggests... yes... because we are in log-land, that weight scales with $height^2$.

One way to 'test' this is to ask whether estimating the log(weight) slope improve the fit of the model compared to fixing it at 2. Here are ways: 'by hand' and by a function in the `car` package.

```
# by hand
# we use the offset function to fix the slope of height at 2
mod_Davis_offset <-lm(log(weight) ~  sex + offset(2*log(height)), data=Davis)

# then compare it to the model where we estimate it freely.
anova(mod_Davis_log2, mod_Davis_offset)
```

```
## Analysis of Variance Table
##
## Model 1: log(weight) ~ log(height) + sex
## Model 2: log(weight) ~ sex + offset(2 * log(height))
##   Res.Df    RSS Df  Sum of Sq     F Pr(>F)
## 1    197 2.5718
## 2    198 2.5726 -1 -0.0007962 0.061 0.8052
```

```
# by car package
car::linearHypothesis(mod_Davis_log2, "log(height) = 2") # the estimate IS 2
```

```
## Linear hypothesis test
##
## Hypothesis:
## log(height) = 2
##
## Model 1: restricted model
## Model 2: log(weight) ~ log(height) + sex
##
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1    198 2.5726
## 2    197 2.5718  1 0.0007962 0.061 0.8052
```

```
car::linearHypothesis(mod_Davis_log2, "log(height) = 0") # the estimate IS NOT 0.
```

```
## Linear hypothesis test
##
```

```
## Hypothesis:
## log(height) = 0
##
## Model 1: restricted model
## Model 2: log(weight) ~ log(height) + sex
##
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    198 3.6131
## 2    197 2.5718  1    1.0413 79.768 2.99e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

10.4 Some General Principles

10.4.1 ANCOVA always tests the interaction against the additive model

This figure highlights the various models and interpretations associated with the pieces of an ANCOVA dataset. This is a way to envision all of the potential linear relationships among the variables that might arise in an experiment with a continuous and categorical explanatory variable.

Note that just because your data looks like a particular pattern doesn't mean you should use that model. Your statistical model should be driven by your question and original design.

For example, if you design a study with an ANCOVA question in mind, you are always starting your analysis with the question of whether the effect of X on Y varies by Z. Thus, ANCOVA always starts with comparing the interaction (model E) against the additive model (model D).

> Proof: If we compare the additive model and the model with the interaction, we are specifically testing for whether allowing the slopes to vary by sex *explains additional variation* above and beyond the main, additive effects of height and sex. To do this, we use `anova()` but pass it two models. This does what is called a Liklihood Ratio Test. Compare the result of the first test to the last line of the second. This proves that when we fit model E, the test for the interaction is a comparison with model D!

```
anova(mod_Davis_log, mod_Davis_log2)
```

```
## Analysis of Variance Table
##
```

	VERBAL HYPOTHESIS	INTERCEPT/ SLOPE	MODEL IN R
A	The number of eggs produced by limpets does not vary with density or season	Common intercept Zero slope(s)	lm(Eggs~1, data=limp)
B	The number of eggs produced by limpets does not vary with density but is reduced in the summer season (dashed)	Different intercepts Zero slope(s) Parallel horizontal line	lm(Eggs~Season, data=lim
C	The number of eggs produced declines with density, but the maximum number of eggs (intercept) and the rate (slope) do not vary with season	Same intercept Same slope Same lines	lm(Eggs~Density, data=lim
D	The number of eggs produced declines with density and the number of eggs at density zero (intercept) differs between seasons but the rate (slope) does not vary with season	Different intercepts Same (negative) slope Parallel lines	lm(Eggs~Density+Season data=limp)
E	The number of eggs at density zero (intercept) and the rate (slope) vary with season	Different intercepts Different slopes 'Crossing' lines	lm(Eggs~Density*Season data=limp)

Figure 10.1: From Getting Started with R, 2nd Edition, OUP

```
## Model 1: log(weight) ~ log(height) * sex
## Model 2: log(weight) ~ log(height) + sex
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    196 2.5574
## 2    197 2.5718 -1 -0.014402 1.1038 0.2947
```

```
Anova(mod_Davis_log)
```

```
## Anova Table (Type II tests)
##
## Response: log(weight)
##                  Sum Sq  Df F value    Pr(>F)
## log(height)     1.04135   1 79.8099 3.024e-16 ***
## sex             0.34461   1 26.4115 6.647e-07 ***
## log(height):sex 0.01440   1  1.1038    0.2947
## Residuals       2.55738 196
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

10.4.2 Specifying models in R

We can add terms `weight ~ sex + height`

Have crossed terms `weight ~ sex * height` which is `weight ~ sex + height + sex : height`

Remove terms `weight ~ sex * height - sex : height` which is `weight~ sex +  height`

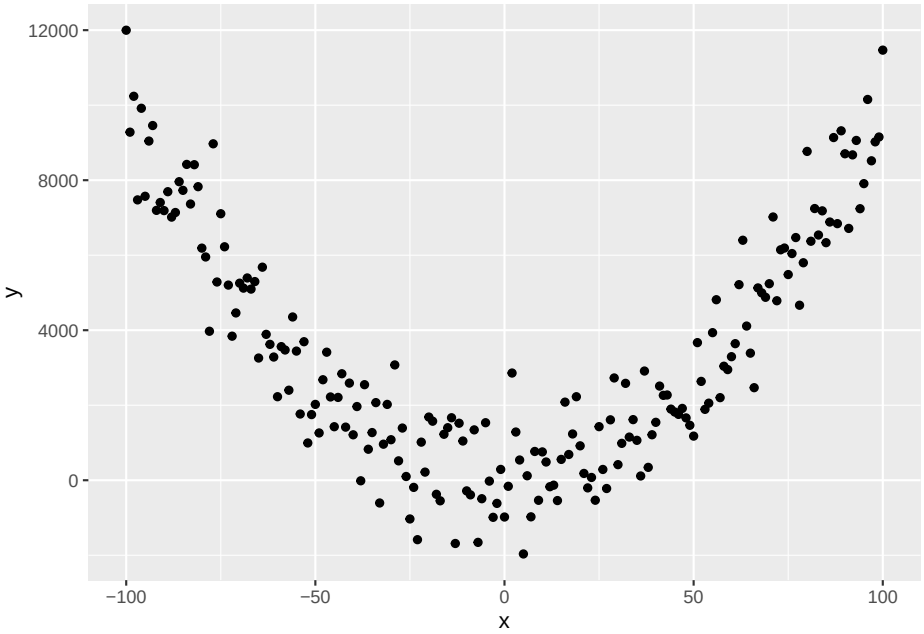
Include all interactions up to order k `weight~ (sex + height + age)^2` which is `sex + height + age + sex : height + sex : age + age : height`

Chapter 11

Nonlinearities, polynomials and regression

If you recall from chapter 1, we introduced the general linear model, and finished with an example where the data were highly non-linear and based on the following equation

$y = 0.01 + x^2 + \epsilon$



In this final section of the module, we are going to analyse a nonlinear relationship using a linear model. This exercise should accomplish two tasks. First,

it will show you how to build models to capture non-linearity. Second, it will reinforce how we use the diagnostic tools to critique our model before moving on to make inference.

11.1 Transformations - a reminder and starting point.

Transformations, as we discussed in previous chapters, allow lots of models to be expressed as a linear model.

For example the model for exponential growth (Numbers vs Time)

$N_t = N_0 * e^{(r*t)}$

Remembering some rules of logrithms, this can be expressed as a linear model as:

$log(N_t) = log(N_0) + r * t$

Here, $log(N_0)$ is the y-intercept and r is the gradient or slope, and t = time is the x-axis explanatory variable and $log(N)$ is the y-axis response variable.

11.2 Working through an example.

Don't forget that you need the `tidyverse`, `car` and `ggfortify` packages (use `library()`) to make all of this work!

We are going to work with a built in dataset about cars to demonstrate how to detect and model non-linearities. This is an exercise in multiple regression and in using the diagnostic plots.

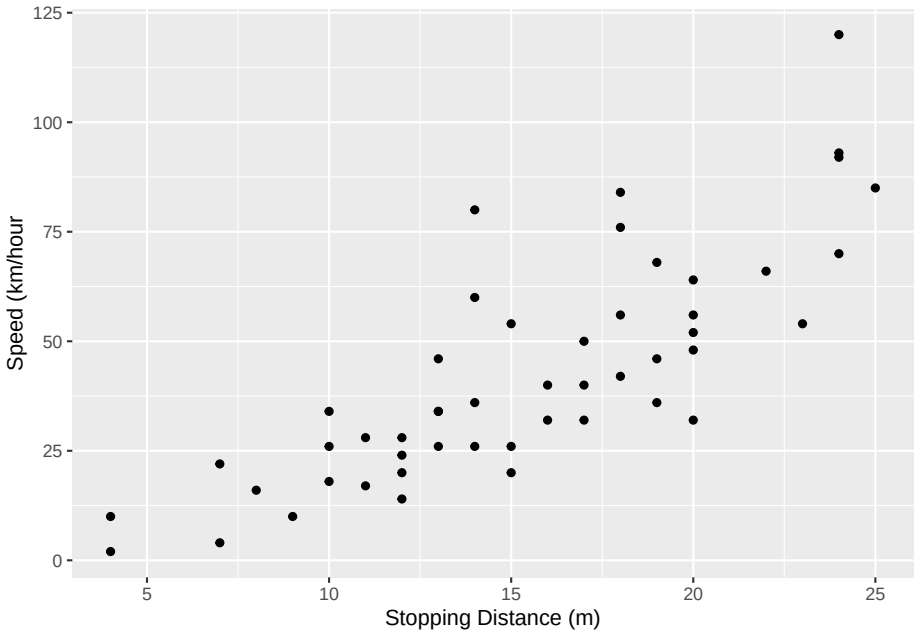
The data are built into R and we don't need to do anything but remember that the dataset is called cars. Lets see what it looks like. The `speed` is how fast the car is going before it is 'asked' to stop, and the `dist` variable is how long it takes to stop in m

```
glimpse(cars)
```

```
## Rows: 50
## Columns: 2
## $ speed <dbl> 4, 4, 7, 7, 8, 9, 10, 10, 10, 11, 11, 12, 12, 12, 12, 13, 13,...
## $ dist  <dbl> 2, 10, 4, 22, 16, 10, 18, 26, 34, 17, 28, 14, 20, 24, 28, 26,...
```

Let's start with step 1: making a good picture.

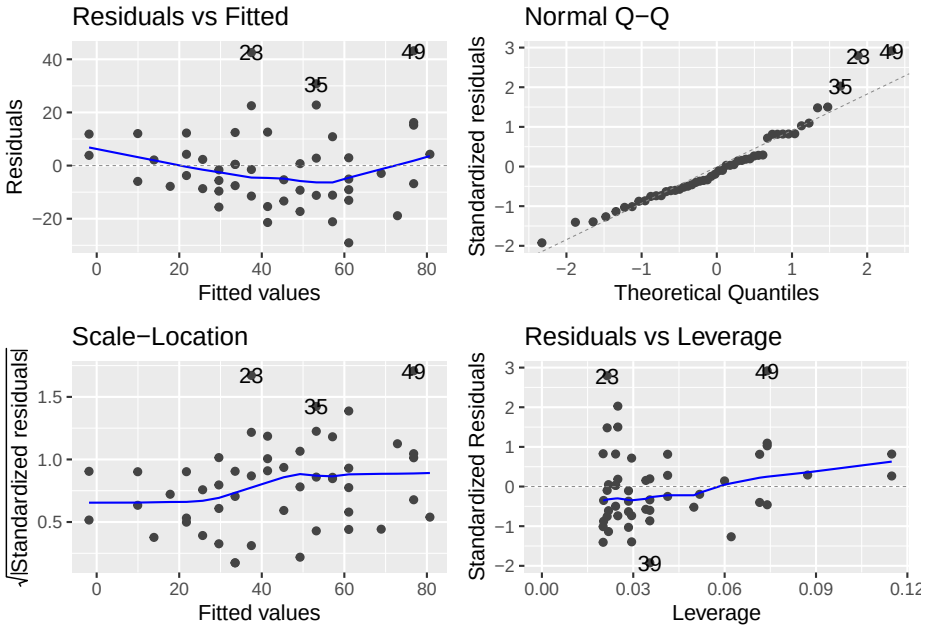
```
ggplot(cars, aes(x = speed, y = dist))+
  geom_point()+
  ylab("Speed (km/hour")+xlab("Stopping Distance (m)")
```



There are a three things we should immediately notice in this figure. The first is that there is a positive relationship. The second is that it appears to be a bit non-linear, even *exponential*. Third, it looks like there is more variation out at high speeds than at low speeds.

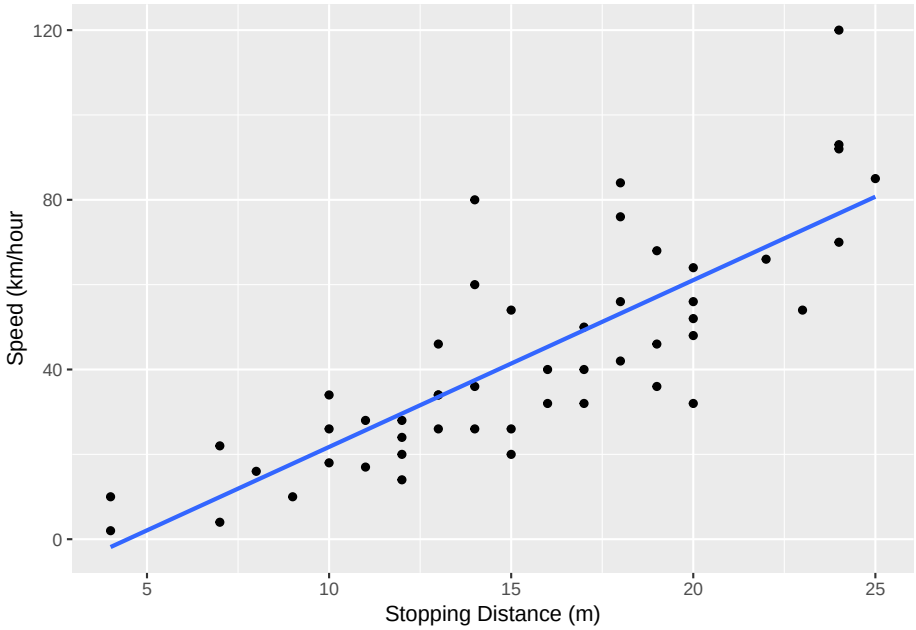
Let's ignore all of this for the moment and fit the model and acquire the diagnostics.

```
mod_dist <- lm(dist ~ speed, data = cars)
autoplot(mod_dist)
```



OK - our preliminary assessment about non-linearity and non-constant variance appears to be spot on! There are systematic departures from the linear model indicated by the upper left plot.

Here we add a line to the data (plot) to understand why the residuals in the upper-left panel is u-shaped. Make sure you get the idea that the majority of the **data** are *above* then *below* then *above* the line as you move across the graph. This generates residuals that have the pattern we see!



The lower left panel of the residuals/diagnostics indicates that the variance does increase with the mean.

11.2.1 Interlude 1 - Inference before fixing the problem

I want to have us look at this model now before moving on. If we ignore the residuals, we end up with a strong conclusion that there is a positive linear relationship between stopping distance and speed.

```
anova(mod_dist)
```

```
## Analysis of Variance Table
##
## Response: dist
##           Df Sum Sq Mean Sq F value   Pr(>F)
## speed      1  21186 21185.5  89.567 1.49e-12 ***
## Residuals 48  11354   236.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(mod_dist)
```

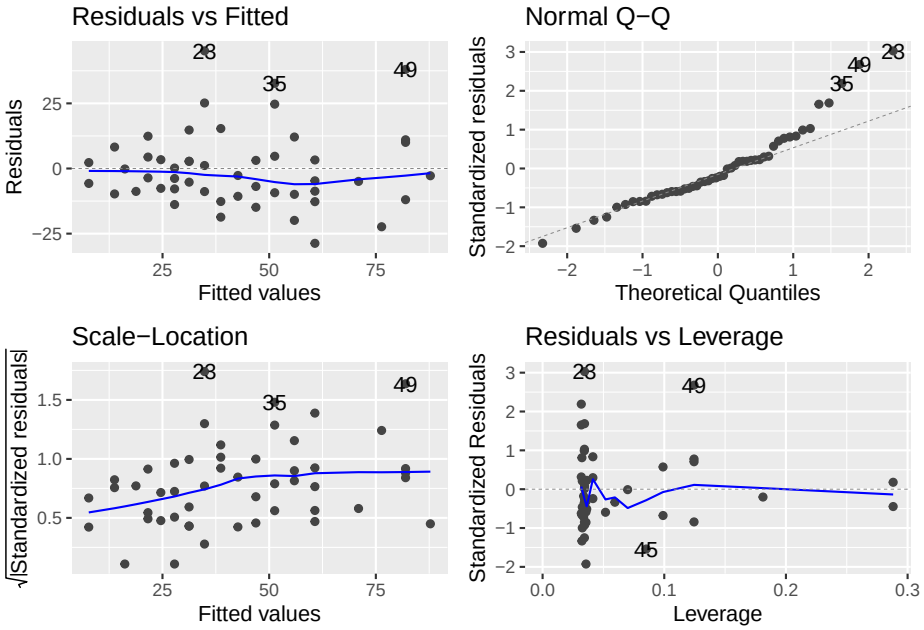
```
##
```

```
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

11.2.2 Solving the Problem I: The Non-linearity

In order to fix this problem, we can take note from our first example in chapter 1 and fit a multiple regression model with a polynomial!

```
mod_dist_nonLin <- lm(dist ~ speed + I(speed^2), data = cars)
autoplot(mod_dist_nonLin)
```



Wow. That's a massive improvement on the diagnostic that indicated a systematic departure from the linear relationship between dist and speed. But.... we still have non-constant variance - the variation in predicted distance is still increasing with the mean.

11.2.3 Interlude 2 - Inference before fixing both problems

As above, lets look at what the model tells us:

```
anova(mod_dist_nonLin)
```

```
## Analysis of Variance Table
##
## Response: dist
##             Df  Sum Sq Mean Sq F value    Pr(>F)
## speed        1 21185.5 21185.5  91.986 1.211e-12 ***
## I(speed^2)   1   528.8   528.8   2.296    0.1364
## Residuals   47 10824.7   230.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(mod_dist_nonLin)
```

```
##
```

```
## Call:
## lm(formula = dist ~ speed + I(speed^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.720  -9.184  -3.188   4.628  45.152
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.47014   14.81716   0.167    0.868
## speed        0.91329    2.03422   0.449    0.656
## I(speed^2)   0.09996    0.06597   1.515    0.136
##
## Residual standard error: 15.18 on 47 degrees of freedom
## Multiple R-squared:  0.6673, Adjusted R-squared:  0.6532
## F-statistic: 47.14 on 2 and 47 DF,  p-value: 5.852e-12
```

Hmm. That's interesting. We've added a term to the model to capture that hint of the non-linearity. There are two ways we might think this is 'unjustified'. First, there is no evidence that this new term is significant. Second, we can formally compare the two models, giving us a p-value that's familiar from the tables above.

```
anova(mod_dist_nonLin, mod_dist)
```

```
## Analysis of Variance Table
##
## Model 1: dist ~ speed + I(speed^2)
## Model 2: dist ~ speed
##   Res.Df   RSS Df Sum of Sq     F Pr(>F)
## 1     47 10825
## 2     48 11354 -1   -528.81 2.296 0.1364
```

One might stop here and return to the data and the diagnostics and thing, hmmm… that systematic departure issue? Not really a big deal after all…. but lets carry on.

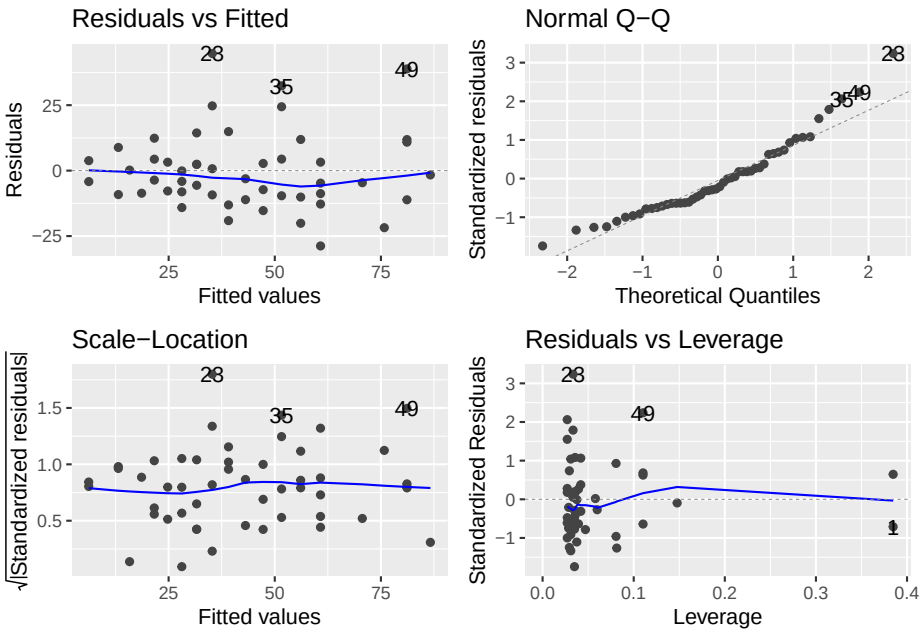
11.2.4 Solving the Problem II: Non-constant - increasing variance

One of the ways to 'fix' this problem, is to make an assumption that the variance is proportional to the mean.

In the absence of this, the relationship is $\sigma^2 * speed$ - as speed increases, the variance increases. But if we assume that the variance is proportional, we get $1/speed * \sigma^2 * speed$…..

To do this in the model, we can use the *weights* argument:

```
mod_dist_nonLin_weight <- lm(dist ~ speed +I(speed^2), weights = 1/speed, data = cars)
autoplot(mod_dist_nonLin_weight)
```



Super. We now have a model where we capture the non-linearity using a polynomial and capture the issue of increasing variance by making the model reflect our assumption that different observations have different variances.

11.2.5 Interlude 3 - Inference with both things corrected.

OK. Now we are ready for using the anova table and summary table to make inference and update our figure.

11.3 Anova Tables with multiple explantory variables.

In previous exercises, we've used `anova()` to generate the anova table, F-statistics and p-values to test our hypotheses. It's very important to note that `anova()` makes a sequential sums of square calculation.

In our ANCOVA example, the model we fit was `lm(log(weight) ~ log(height) * sex, data = Davis)`. We know this expanded to `log(height) + sex + log(height):sex`.

When we reviewed the Anova Table, there were four rows corresponding to each of these three terms (2 main effects and 1 interaction) and the residual (remaining) variation.

In this table produced by `anova()`, we read it like this. First, we ask how much variation is explained by log(height). Having explained this variation, and left some unexplained, how much is then explained by sex. Finally, having explained some variation by log(height) and some of the remaining then by sex, can we explain any more by allowing slopes to vary (the interaction).

The key thing to note here is that this was a sequence. Sometimes this is fine - all we are interested in is the last line, for example, testing the hypothesis that the effect of height on weight varied by sex. This makes sense when we've designed an experiment to test a hypothesis.

However, in multiple regression, like our example with speed and distance, we need to worry about the order. It's really important too worry about this when we've NOT done a manipulative and designed experiment. This is because the terms in the model can be collinear. In an experiment, treatments are always independent. In observations studies, or when we start making polynomials, variables become non-independent.

Collinearity occurs when:

-One of the predictor variables is a function of another as in speed and speed^2
-Your design is unbalanced, generating a non-orthogonal design – note not all unbalanced designs are non-orthogonal.

How do we avoid using the sequence? Well, let's introduce the trick, and then reveal how it's working.

The trick is `Anova()` - not the big A. We call it **Big-A Anova**. It's in the `car` package.

```
Anova(mod_dist_nonLin_weight)
```

```
## Anova Table (Type II tests)
##
## Response: dist
##             Sum Sq Df F value Pr(>F)
## speed        12.72  1  0.9009 0.3474
## I(speed^2)   34.44  1  2.4402 0.1250
## Residuals   663.42 47
```

11.3.1 Explaining the Trick

So, this table is actually constructed by fitting two different models. One of those models is where *speed* is first and $speed^2$ is second. The other model is where $speed^2$ is first and *speed* is second.

```
anova(lm(dist ~ speed +I(speed^2), weights = 1/speed, data = cars))
```

```
## Analysis of Variance Table
##
## Response: dist
##            Df  Sum Sq Mean Sq  F value    Pr(>F)
## speed       1 1609.18 1609.18 114.0024 3.704e-14 ***
## I(speed^2)  1   34.44   34.44   2.4402     0.125
## Residuals  47  663.42   14.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lm(dist ~ I(speed^2)+speed, weights = 1/speed, data = cars))
```

```
## Analysis of Variance Table
##
## Response: dist
##            Df  Sum Sq Mean Sq  F value    Pr(>F)
## I(speed^2)  1 1630.91 1630.91 115.5417 2.957e-14 ***
## speed       1   12.72   12.72   0.9009    0.3474
## Residuals  47  663.42   14.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pay close attention…. can you see which of the *speed* and $speed^2$ F and p-values from EACH of the two models are combined in the Big-A Anova table? It's from when the are *last in the model*.

Technically, this is called a Type II anova table. Technically, it means that we are estimating the effects of each term, controlling for the other and respecting 'marginality'. The Big-A Anova takes care of this *'rotation of the position of the terms'* for us and produces the accurate test table we need.

For unbalanced data Type II ANOVA is recommended. In many cases the biology can be used to settle the question. Say for example if you're interested in the effect of a flower type on pollination after correcting for the effects of plant size - you know plant size affects the number of pollinator visits. Or if you have blocks (what are they?) then you might be interested in treatment affects after correcting for the effect of block.

11.4 Types of ANOVA

There are 3 types of ANOVA model based on how the sums of squares are divided. They hinge a bit on understanding **The Principle of Marginality** :

-A model with the A:B interaction must contain both the A and B main effects. -The main effects are said to be marginal to the interaction. -More generally if a higher-level interaction is included then all lower level interactions and main effects must be included. -See Hector et al. 2010 Analysis of variance with unbalanced data: an update for ecology & evolution. J Anim Ecol for a clear discussion of the different approaches.

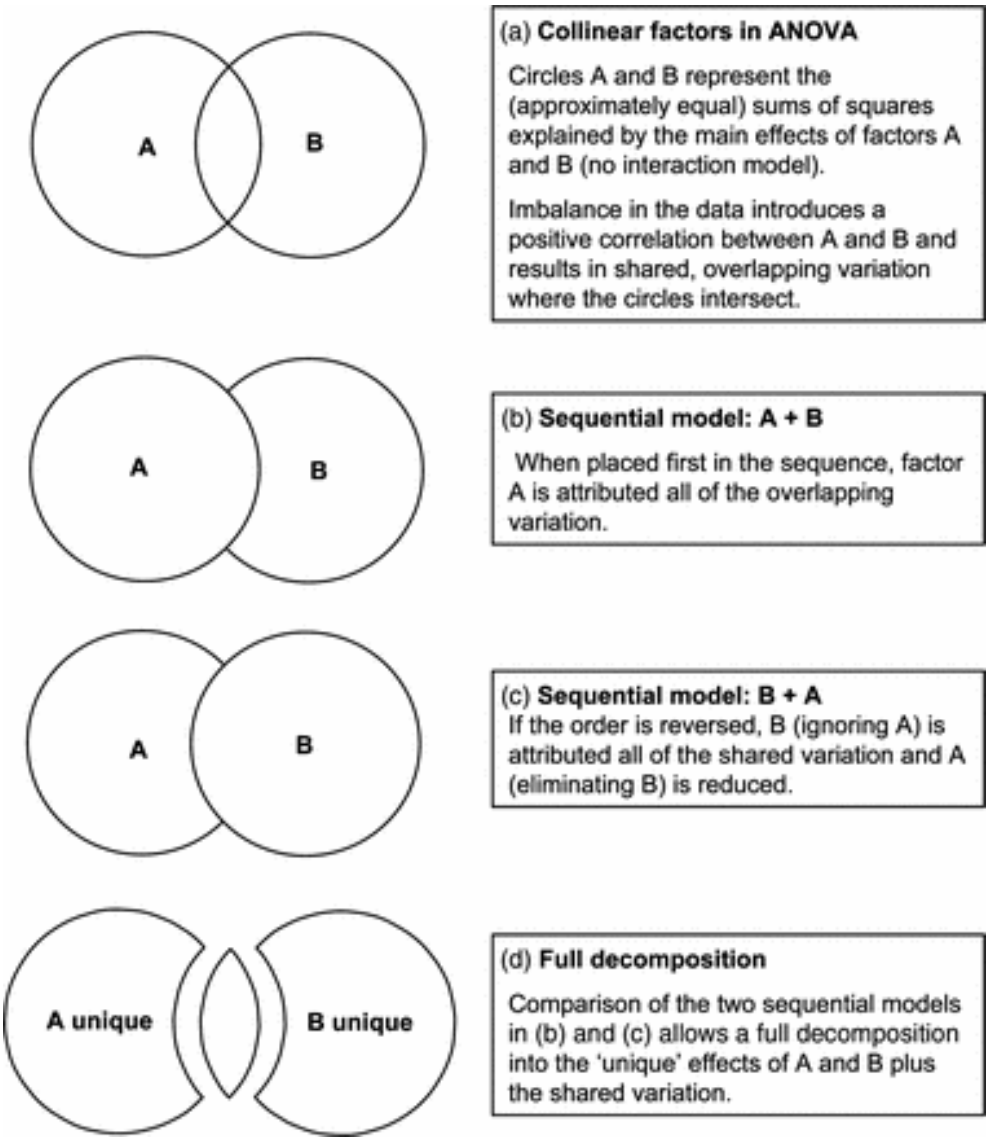
The types of ANOVA:

Type I – this is a sequential anova so order matters – not very useful with unbalanced data or correlated predictors. Just fine for manipulative experiments and specific hypotheses captured by the 'last term in the model'.

Type II – measures effect of a predictor with all other predictors in the model, but obeys marginality principle. More sensible and conservative. A good standard.

Type III – can test main effects after interactions, so test the main effect of A after the main effect of B and the A:B interaction ignores marginality principle.

This diagram may be valuable in helping you understand these ideas.



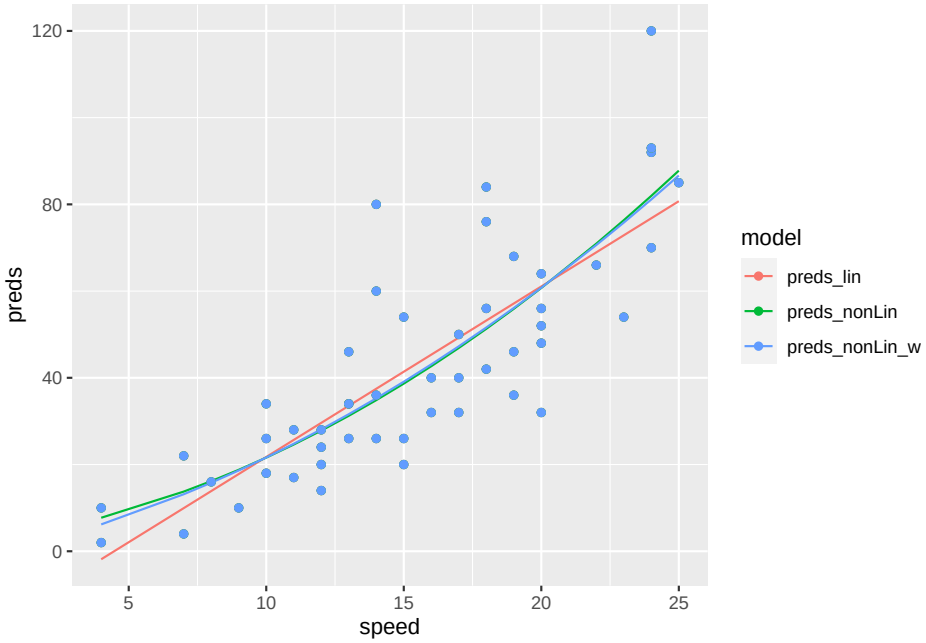
11.5 Explaining the result - a dose of reality

So, hopefully that makes sense. We'll review more of that in the generalised linear model module with Dylan Childs.

What you might be wondering, however, is what the result means. We have no p-values that are < 0.05. Controlling for speed, there is no effect of speed^2 and vice-versa. We kind of new that from our initial 'sequential' assessment above….

OK. Whoa. This is weird, right?

Let's plot all of these three models we've made together with the data



At this point, we should ask, is the non-linear model makeing sense? We think it probably should, right? It's what the data look like and there was a hint of it in the residuals. But in the end, there is no evidence for a relationship.

Perhaps there is a better way to go about this. And in this better way a lesson in p-values and statistics.

> Nothing beats a good question and an understanding of the mechanism you are actually evlauating.

11.6 A mechanistic model

Here we have a mechanistic model stopping distance in made of:

- Thinking distance – distance travelled while driver reacts so this is reaction time * speed
- Braking distance – which depends on the car's kinetic energy which is proportional to speed^2

So we expect a model of the form – note that there is NO intercept

$Distance = \beta_1 * Speed + \beta_2 * speed^2$

We can fit this model really simply with that trick of -1 …. and all of a sudden….

```
# fit model without intercept
mechMod <- lm(dist ~ speed +I(speed^2) - 1, weights = 1/speed, data = cars)

# all the inferences
anova(mechMod)
```

```
## Analysis of Variance Table
##
## Response: dist
##            Df Sum Sq Mean Sq  F value    Pr(>F)
## speed       1 5997.7  5997.7 433.8850 < 2.2e-16 ***
## I(speed^2)  1  137.1   137.1   9.9154  0.002818 **
## Residuals  48  663.5    13.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova(mechMod)
```

```
## Anova Table (Type II tests)
##
## Response: dist
##            Sum Sq Df F value   Pr(>F)
## speed       92.60  1  6.6990 0.012727 *
## I(speed^2) 137.06  1  9.9154 0.002818 **
## Residuals  663.51 48
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

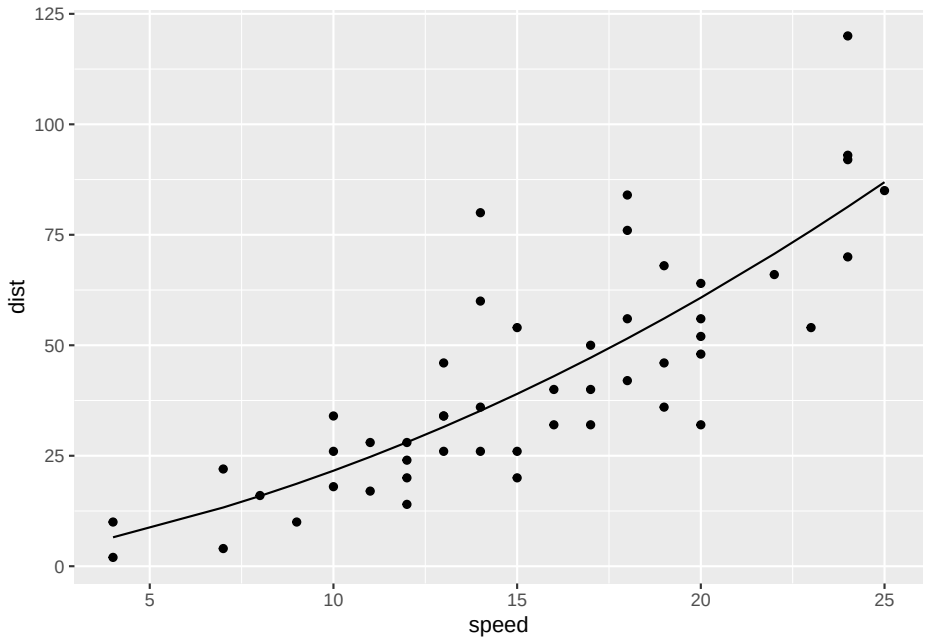
```
summary(mechMod)
```

```
##
## Call:
## lm(formula = dist ~ speed + I(speed^2) - 1, data = cars, weights = 1/speed)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6.4315 -2.3314 -0.8756  1.6428 11.9808
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
```

```
## speed       1.28525    0.49657   2.588  0.01273 *
## I(speed^2)  0.08764    0.02783   3.149  0.00282 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.718 on 48 degrees of freedom
## Multiple R-squared:  0.9024, Adjusted R-squared:  0.8983
## F-statistic: 221.9 on 2 and 48 DF,  p-value: < 2.2e-16
```

```
# plotting
plotMech <- cars %>%
  mutate(preds = predict(mechMod))

ggplot(plotMech, aes(x = speed, y = dist))+
  geom_point()+
  geom_line(aes(y = preds))
```



11.7 The Take Home Messages

This example is not out of the ordinary of what you might face with your own data. The recipe for success in data analysis is a bit different from what you might expect.

1. Design experiments to test hypotheses with Randomisation, Replication and Reducing Error in mind.
2. Think about your question and the mechanisms expected to give rise to the pattern
3. Plot your data and look at the patterns.
4. Fit appropriate models. Models for experiments are easier to define than models from survey and observation. Models for experiments don't have to worry as much about collinearity….
5. Don't follow the p-value.
6. Eat Biscuits

Chapter 12

A final Overview