Statistics for Psychologists

John Towse, Tom Beesley, Margriet Groen, Rob Davies

2021-11-11

Contents

Chapter 1

Intro

This is a collection of tuition material written for Psychology undergraduates at Lancaster University. At the moment the content represents the "lab materials" for the PSYC121 and PSYC122 modules in first year. They feature tuition of programming with R, with material designed to be accessible for students without any programming background.

1.1 Analysis labs and 'pre-lab work'

Where the 121 Analysis lab includes a "pre-lab" section – such as this one! – please ensure you have (a) watched the relevant pre-recorded lecture material (b) read through *this section* before your assigned lab class session and (c) completed any specified exercises.

The lecture is designed to deliver important ideas and procedures for learning about analysis. Pre-lab material is then designed to help consolidate this learning, or enhance, expand and apply it in ways that set the scene for the lab session activity. We want to prepare you to be ready to go in the session itself and make the most of your time there.

> Top tip for part 1: Going into sessions prepared will help you get more out of the teaching.'
>
> Getting an overview or briefing on what you will be doing by reading relevant material etc, means you know what to expect, and where to focus your energy. Don't be shy about coming with questions!

1.2 Computing systems

For all University systems you will need to be a registered user and have received a 'username' and details about how to generate your 'password'. If you are struggling to set this up, then contact ISS ASAP.

All staff and students have e-mail accounts on the University system. Normally an e-mail address takes the form: first initial.surname@lancaster.ac.uk

For example, to contact the Content Director of PSYC121, Tom Beesley, you use t.beesley@lancaster.ac.uk (usually you can also use the shortened version, t.beesley@lancs.ac.uk)

When the department needs to contact you individually we send a message to your **university e-mail account**. In addition, the University e-mail security system, specifically the SPAM prevention procedures, will frequently "junk" e-mail from external accounts from free systems (e.g., hotmail, google, etc.). It's always preferable to use your University email account so staff know it is a genuine student request.

Consequently we stress that you:

- DO NOT use external accounts when contacting the Department or Staff as messages from these are not guaranteed to be delivered (and are routinely not replied to by Staff)
- Check your university account daily, as important University and Departmental messages may be in this mailbox

Moodle

Moodle plays a key role in your time at Lancaster. Moodle is the University VLE (Virtual Learning Environment). Moodle is used to:

- Communicate course information and urgent messages to all students on a course. As a result **it is essential for all students to log in at least once a day**
- Make course materials, such as lectures, lecture slides/notes, and lab handouts, accessible to students prior to these sessions
- Provide the web links for each Web-Based Assessment (WBA)
- Provide forums for online discussions regarding course material (students are encouraged to use these forums to raise questions about course material that they seek clarity on)

1.3 Other Department research systems

The Department SONA system

In your full online Part I handbook you can read about the Research Participation Scheme (run through the SONA system). This scheme offers students the opportunity to take part in research studies. The studies cover a range of research topics and are interesting to take part in. They provide an invaluable learning opportunity for students in terms of how to design experiments, how to treat participants in an ethical manner and how to run an experiment. Many research studies in the first term will be carried out by final-year undergraduates collecting data for their research projects. This may well be you in a couple of years time!

Psychology students registered on PSYC121 are strongly encouraged to take part in these studies and collect a minimum of 20 credits* [* this is an estimate at the time of writing - the target will be communicated when confirmed within the department]. There are multiple incentives.

First, completion of these credits is an essential pre-requisite to use the SONA system to recruit participants to your own study when you enter your third year. (nb you are required to carry out your own research project as part of your Psychology degree). This benefit of this access should NOT be underestimated.

Second, once you have collected 20 credits, you will be able to collect additional credits, and earn further project credits for yourself in the third year. But also, taking part in studies, run by other undergraduates, or postgraduates or Research Assistants or staff, is a fantastic way to learn about research methods and experimental designs and psychology in general. Time and again, many students tell us that they got great ideas for projects and for good experiments from their experiences as participants. Being a research participant provides unique insights into what goes on in experiments and what we can learn from them.

You will receive more information about SONA – such as how to register to gain access to studies – in the next couple of weeks.

Chapter 2

Week 1: Introduction to Statistics with R Studio

Written by John Towse & Tom Beesley

The exercises in this section are designed to familiarise you with working in R Studio.

2.1 Lab Work

2.1.1 Introducing R Studio

R and R Studio is the software that we will be using to explore and learn about analysis in your Psychology degree. It's a computational engine: a very snazzy calculator that you should see as your friend and ally in the journey to understand and appreciate psychology. It sits *alongside* what we teach about the concepts and interpretation of statistical analysis.

R is the core software, R Studio is the interface for interacting with it. Put another way, *R is the engine, R Studio is the cockpit.*

Like even a simplest calculator, it just does what you ask (at least when you ask nicely!) but it requires the user to know what they want from it and to understand what it is telling you. A calculator can't help a kid get the right answer to a multiplication problem if they don't know the difference between multiplication and division and addition etc. And whilst a calculator is brilliant at doing the number crunching (and as a bonus, R Studio can help with turning the numbers into beautiful graphs and images too), even a calculator requires

a thoughtful person to take the answers and make sensible interpretations from them.

Therefore, we need to learn both about the concepts of statistical analysis on the one hand, and the processing of statistical information -through R- on the other. The lectures will provide the starting point and the direction for statistical concepts, whilst these analysis labs provide the more practical experiences in how to use R, and how to R your ally. Over the next year, in these labs we will increasingly be using R Studio to focus on the latter, processing side, which will allow you to focus your energies on the conceptual side and its relevance for appreciating psychology.

2.1.2 Getting started with R Studio

For Lancaster University Psychology Students in 2021, we will be learning about R Studio through a simple but powerful web server architecture. That is, through the power of the internet, you can access and use R Studio by logging into a free account that we have provided and we will maintain for your use.

> Here's a little secret: There are several different ways to access R Studio. For example, you can download a copy of the software onto your computer, or use a Virtual Machine set up to run a copy. There's nothing to stop you having your local copy, but please note - we can't support your own version through lab classes

So why are we using an R Studio web server?

- Uniformity of experience. It doesn't matter whether you have a PC, a Mac, a Chromebook or whatever. The server provides a consistent, uniform experience. That also means we can spend more time helping you master the software, rather than translating between versions on different platforms
- Ubiquity of experience. There are **lots** of computers on campus. There are computer rooms, there are pods in the learning centre, there are bookable laptops in the Levy Lab, and so on. Providing a web interface from a browser, means you can practice R from all the different machines. Your not limited to a particular copy on a single machine.
- Consistency of performance. Since everyone logs into the same installation of R Studio, we can ensure that things work the way we expect. This allows staff to focus on teaching you the key issues, without worrying about software conflicts because different machines have different configurations, or different versions with different features etc. We'll have more time to focus on the important stuff!

- Quality of performance. Some installations of R Studio and some commercial versions of the server are set up, well, to be the basic version rather than the premium version. To draw a papelle: do you want to have to put up with the cheap, low-bandwidth broadband package? We've got control of the server set up, so we'll try and make this the most effective way to do this.

We've moved to the R Studio server approach this year. We've tried hard to set it up to work well- Apologies in advance for any teething issues, we will try to resolve them as quickly as possible.

You will have received an email with your account information to log onto the R Studio server. Please keep your account details safe.

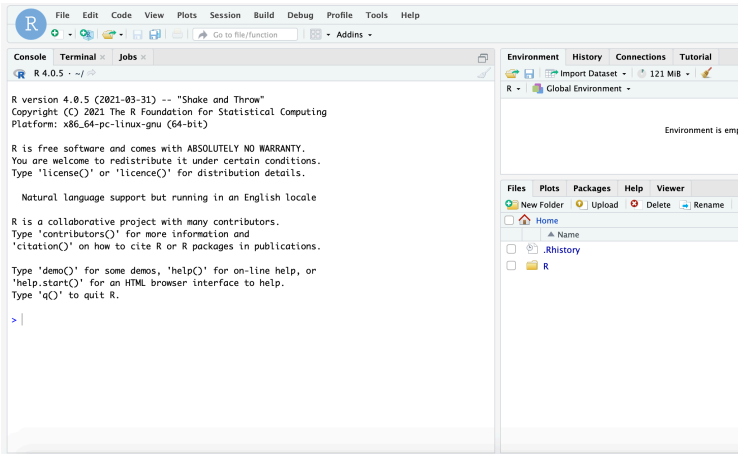
From a computer on the campus wifi, you can access R Studio at:

psy-rstudio.lancaster.ac.uk (off campus, you will need to be on the VPN)

At the login screen, use your university username (e.g., bloggsj)

Your password for R Studio is: [password here]

2.1.2.1 What does it look like?



When RStudio starts, it will look something like this:

RStudio has three panels or windows: there are tabs for Console (taking up the left hand side), Environment (and History top right) , Current file (bottom right). You will also see a 4th window for a script or set of commands you develop, also (on the left hand side).

2.1.3 Let's do something!

In the console window on the left hand side, there's a command prompt ">". This is where we ask RStudio to do our bidding!

1. Click in the console window and we will get R to work as a calculator. Type in:

```
5 + 5
```

and press enter. You should get the answer (amazing huh? OK, maybe not *that* amazing...). Use your imagination – ask a simple arithmetic question of your own choosing!

2. In the first analysis lecture, we looked at measures of central tendency and how to calculate them. So let's get R to do these calculations also!

First, we tell R about the data from the lecture. Copy the following line and paste it into the console, then press enter to run it:

```
PSYC121_week_1_data <- c(7,8,8,7,3,1,6,9,3,8)
```

This creates an "object" called `Analysis_week1_data`. We can then perform calculations on this object. For example, we can find the mean by using the following command (again, copy and paste)

```
mean(PSYC121_week_1_data)
```

Check the answer is the same we found in the lecture (it should be 6!).

Next, let's ask for the median:

```
median(PSYC121_week_1_data)
```

This also should be the answer from the lecture (7)

R doesn't have a single corresponding command for the *mode*, but we can use this series of commands:

```
getmode <-
  function(PSYC121_week_1_dataa) {
    uniqv <- unique(PSYC121_week_1_data)
    uniqv[which.max(tabulate(match(PSYC121_week_1_data, uniqv)))]
    }

getmode(PSYC121_week_1_data)
```

This is just a bit of clever jiggery-pokery that gets the mode.

2.1.4 Extra content

The commands above are designed to show you that with RStudio active, you can get quick and accurate answers to material covered in PSYC121 lectures.

All we have asked is that you write (or copy in) text to get the information. However, after the lab, you could play around with RStudio in your own time and think about the following:

In R, "<-" is the assignment operator as in the command we used:

```
PSYC121_week_1_data <- c(7,8,8,7,3,1,6,9,3,8)
```

We create the variable label on the left (`Analysis_week1_data`) and we give it those number on the right. The name `Analysis_week1_data` is largely arbitrary: try use a variable of your own naming (your own name?) instead - and then use that alternative name for the other commands.

Throughout this year, we'll use the convention of the "underscore" to separate words in labels (it_makes_them_easier_to_read than ifyoudidn'thaveanyspaces)

What does that tell you about the text used to get the mode? Can you figure out what each line does?

Also, once you have created a variable, you can check what the variable comprises by calling it at the command line. Just write its name, and R will respond with all the data points (all the X values it knows about). Take note that, as you write the variable label, R studio should offer to "auto-complete" the name. It only does that if it you have defined the variable, or are writing a known command.

Chapter 3

Week 2: Descriptive statistics in R Studio*

Written by John Towse & Tom Beesley

3.1 Pre-lab work

Last week we asked you to

- Practice a WBA task (the math skill task)
- Connect to the R Studio server and enter some simple commands
- Complete a survey so that we can collect data for analysis teaching

This week - we are introducing the `learnr` tutorial system for lab preparation. This will allow you to try out some useful lab skills before your main lab.

Please make sure you've looked at and followed through the exercises here: https://ma-rconnect.lancs.ac.uk/PSYC121_W2_labprep/

3.1.1 File organisation

We also recommend that you set up your University "H drive" by following the instructions here

If you do this, you can organise your files in one place and access them from any university computer that you log in to, including the lab machines in the Levy Lab and the virtual machine (see above). So this effectively acts as a free, cloud-based storage solution for your work (and even personal stuff).

However you choose to store your files, it's essential that you are careful about file organisation. As you progress through these exercises, you will find yourself downloading many data files, and creating many R scripts.

Consider each week a new set of materials that should have it's own folder. Get used to organising your work and you will be well placed to revise the material at a later date.

3.1.2 R Studio tasks

For a reminder of how to start R Studio, see the Week 1 content.

You can access the RStudio Server at: http://psy-rstudio.lancaster.ac.uk/

(remember: off campus, you will need to be on the VPN)

3.1.3 Bringing data, scripts and files into R Studio

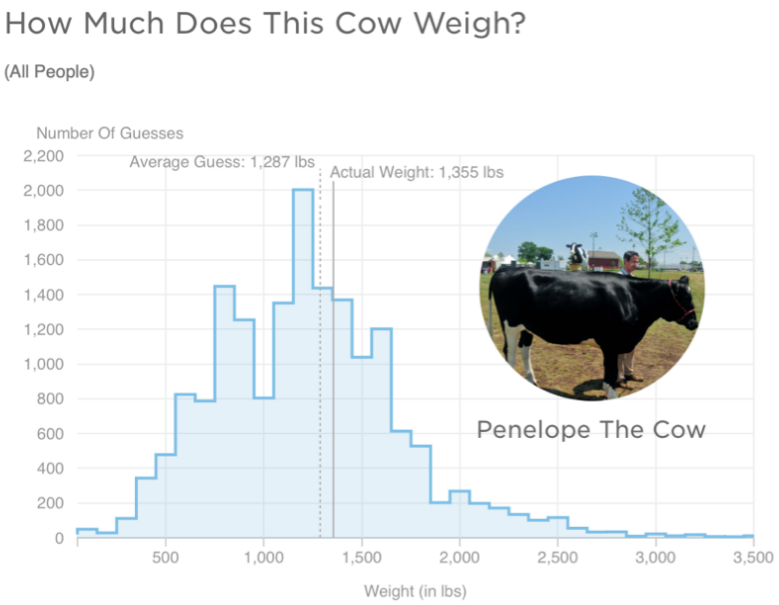
In week 1, we had a tiny dataset that we entered into R through command line. We're going to need a way to tell R about bigger datasets though, so we need to work with datafiles, and bring them into R so that they can form data frames and other forms of data objects in R.

We've provided (on moodle) a walk-though of how to get a data file (in a zip compressed file) into the R environment. Note, once we have imported it into R, it's no longer the original file started with, but within R is has become a named data frame.

3.2 Lab exercises

3.2.1 Descriptive information in R Studio

Some 5 years ago, a large group of participants gave an estimate of the weight of Penelope the cow. Just over 17,000 guesses. And the distribution of guesses was



something like this:

What we can see from this graph is that:

1. Guesses formed a roughly normal distribution. There is a bit of a skew with a right-hand tail, but this is inevitable as a weight of less than 0 is physically impossible, but there is no limit of the semantics of a large guess.
2. The mean guess weight (1,287 lbs) is very close to the actual (true) weight of the cow (1,355 lbs). So even though lots of people were inaccurate, a central tendency measure has a pretty good alignment with te true weight. This is known as the Wisdom of Crowds phenomenon, first identified by Galton in 1907 (though he suggested using the median weight)

Let's look at (a smaple of) the PSYC121 student data from 2021 collected on guessing the weight of Penelope, and ask whether it resembles the properties of this large dataset.

3.2.2 Loading the data

To get the data into R Studio, you need to complete the following steps:

1. Download the "zip" file by clicking this link
2. Find the location of the file on your computer and check it is saved as a ".zip" file
3. Return to RStudio and navigate to the folder you want to upload the data into (you could create a new folder for this week).

4. Click "Upload"
5. Check the target directory is correct
6. Click choose file and find the file on your computer.
7. Select the file and click "Open". Click "OK"

You should now see the files extracted in the directory. If you receive an "unexpected server error" please try this process in a different browser. If you still have trouble, send your username to us t.beesley@lancaster.ac.uk for support.

3.2.3 Using the R script

Let's start working with our data, by opening up the "Week_2_script.R" file. It's part of the zip file for week 2. This is an "R script" that contains a series of commands. We could type each of these in the console window (like we did in Week 1), but by having them in a script like this, it saves us typing them or copying and pasting them into the console. It also means we have a record of all the commands we have run.

The first command is to load a library of functions:

```
library(tidyverse)
```

To run this, simply click anywhere on line 1 of the R script to put the cursor there, and press ctrl+enter (cmd+enter on a mac) or click the button called run. You will see a number of messages appear in the console. Don't worry about these, or worry too much about what exactly this command is doing. Essentially this is giving us some useful tools for our analysis. We will introduce the features of the **tidyverse** gradually during this course.

(side note: If you were using a local version of R studio on your computer, it might not have the 'tidyverse' library already installed. You would need to install the package first)

To import the data into RStudio, from the csv file, simply click the "Import Dataset" button in the "Enviornment" pane. John has a video available to follow along with, if you have trouble with this bit.

The data are in R studio if you have followed all the lab sheet to this point. Note that when you imported the data into the R environment, a command line was generated at the console

```
penelope21 <- read.csv("~/penelope21.csv")
```

What this command accomplished was to read the spreadsheet called 'penelope21' into an object in R called penelope21. You could use any object label, but it's important to then keep that name consistent in what you do next.

The command was also generated

```
View(penelope21)
```

which presents the data in a windo of R studio. Note that "NA" means not available or missing data. Does this file structure make some sense to you?

3.2.4 Finding the mean and median estimates

Use the data to answer the following questions…

1. What is the mean weight estimates?
2. What is the standard deviation of the estimates?
3. What is the median weight of the estimates?
4. Which of these central tendency measures is the more accurate measure of the true cow weight? (make a judgement)
5. What is the mean weight estimate (and standard deviation) for female respondents and non-female (male / non-binary /prefer not to say) respondents?

You may be thinking, how do I possibly do any of this?! Well this week most of the commands you need are contained in the R script you have downloaded. Also, remember from last week, we explored the R command:

```
mean(lecture1_data)
```

That gave us the mean of the small dataset "lecture1_data". This time, we want to explore the penelope dataset. But also, the lecture_data was just a single list of numbers. The penelope21 object is more like a datasheet. So we need to tell R Studio which **column** we are interested in. RStudio uses the format **data$column**. Hence, we can ask

```
mean(penelope21$estimate)
```

(this command is in the r script so you don't need to write it out) and to get a standard deviation we can use the command:

```
sd(penelope21$estimate)
```

So from this, can you work out what you would do to get the median value (remember from last week how we got the median value?). Part fo the command is given to you, can you change the text so that it works?

3.2.5 Calculations from a range of columns

We have seen that:

```
mean(penelope21$estimate)
```

will provide a mean of the column "estimate". In the third column, named "female_estimate", we have the estimates of just the female respondents. In the fourth column, named "other_estimate", we have the estimates of the "other" respondents (males and non-binary and prefer not to say).

So can you now figure out how you might get information about the estimate from the female data (only) or the non-female data? Try it, based on what you have just done. Does it work?

You will find that the result of the this command produces an "NA" result. This means that the answer is "Not Available", or in other words, is a "missing value". This is because some of the values in this column are NA, and the mean of a column with NAs will always lead to the result NA.

Instead, try this command:

```
mean(penelope21$female_estimate, na.rm = TRUE )
```

Any different? The `na.rm = TRUE` instruction tells RStudio that missing data can be ignored in this mean calculation. (in technical language, `na.rm` is a parameter of the function `mean` that removes the NAs if set to TRUE)

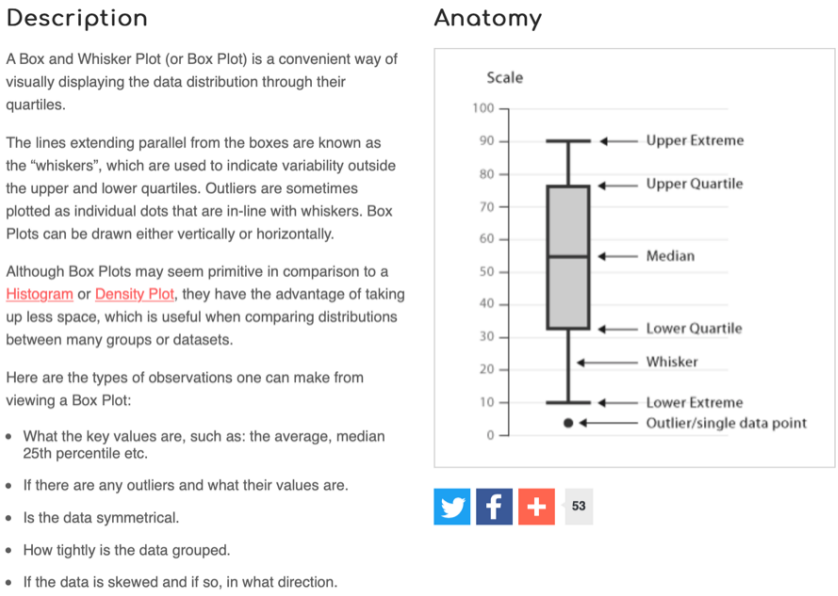
3.2.6 Simple graphs

RStudio can be used to create graphical data plots that can help interpret datasets

The first thing we can do is create a histogram distribution of guesses from the sample student data to compare with the previous large sample study (i.e. the 17,000 guesses):

```
hist(penelope21$estimate)
```

We can also create a "box and whisker plot". Here's a general simple description of a box-and-whisker plot as a graphical representation of data:



We can create a box and whisker plot for the estimate column using the following command:

```
boxplot(penelope21$estimate)
```

3.3 Extra content!

Want to try explore some more outside of the analysis class before next week and improve your R skillset? Try these exercises:

3.3.1 Helping yourself!

If you're not sure how a command works, or some detail of command, then R itself may be able to help you. You can ask for help for example by entering the command

help(mean)

Try get help on other commands we have explored this week. Does the help info converge with your growing knowledge of the commands? Does the help show you some of the additional options with the commands? Can you use the examples which may accompany the help info?

You can also use help() to find out more about packages rather than just commands (see next challenge). And most help info will give some example, but you can ask for examples specifically with example() (e.g. example(mean))

3.3.1.1 Create a box-and-whisker plot of BOTH the female and non-female data on the same figure.

To do so, you can use a really flexible toolkit (in R-speak, called a package) with publisher-quality graphics capabilities called "ggplot2". We turned this on with the command before

```
library(tidyverse)
```

Have a look at the help information about for example, "geom_boxplot", and try out the script information provided for week 2. The challenge is to figure out how the script instructions produce the graph – what do the elements do (maybe fiddle around with it to find out!).

Good luck!

Chapter 4

Week 3: DVs and IVs in R Studio

Written by John Towse & Tom Beesley

4.1 Pre-lab work

Last week we asked you to

- Complete the teaching dataset survey
- Prepare for the lab class with the learnr tutorial
- Complete some R Studio exercises working with data and a script to get some descriptives and some data

This week - again, there's a learnr tutorial to follow and help prep for this week's activties : https://ma-rconnect.lancs.ac.uk/W3_LabPrep/

About 1/3 of students at the time of writing this had yet to complete the teaching survey. We'd really appreciate more completions so we can work with a good sample of data this term, and as a reminder, the data are here: https://lancasteruni.eu.qualtrics.com/jfe/form/SV_6rLPbl1YI8Y2WXj

Since we've got less data than we would expect, we can't yet use the data in this term's activities.

4.2 R Studio tasks

For the "penelope21" data in week 2, we provided you with estimates data, and you were able to generate descriptive statistics for the estimates (if not, please

go back and work through that part of the week 2 lab sheet again). You also found the weight estimates for the female and non-female guesses, right?

However, in order to find the estimates separately for gender identity, we needed to have a column for each gender category. Whilst that worked, it could get cumbersome over time always to work with data created like that.

There is a better way

4.2.1 Task 1 - penelope21 data

Download the week3.zip file and upload it into R Studio server. If you need them, here are the key instructions from Week 2.

Import the "penelope21" data using the `read.csv()` command (if unsure, see here in Week 2), and launch the R script.

This week, we want to explore commands from the tidyverse library (toolkit) which can help us do more powerful things more elegantly. So let's get R to work with the tidyverse library

```
library(tidyverse)
```

(Nb. This command is the accompanying R script. You can access it from there.)

This time, let's ask for the estimate data arranged by identity:

```
aggregate(x = penelope21$estimate, by = list(penelope21$identity),
FUN = mean)
```

First, let's try this! What do you get? Does this match what we did last week when we calculated the mean for the female and for the other group?

Second, let's look at what is happening here:

`aggregate` This is a command to get descriptive statistics Remember we can use help to find more about this command!

`x=`
This defines what column we are analyzing

`by=list`
Now we tell R how to group the estimate data, and which column does that

`FUN=mean`
Specifies which descriptive function is being asked for Can you explore whether you can call on alternate measures?)

4.2.2 group_by()

There's another way that also allows us to group scores by a (nominal) variable. This is explored in the learnr tutorial, which should help you create the command the get weight estimates broken down by gender identity. You need to define the data frame for the estimates data, and the gender IV and the estimates DV

```
*MISSING* %>% group_by(*MISSING*) %>% summarise(mean_estimate =
mean(*MISSING*))
```

First, try this command and see what you get. If you run this command as entered, it won't work. So now use your experience at skills from the above and the learnr tutorial to work out what is required.

Note

`%>%`
This is a "pipe operator", basically take the output from the left and feed it into the requests on the right. `Summarise`
Provide summary statistics information for the specified variable as specified (whether mean, median etc)

4.2.3 The assignment operator

As well as learning about the pipe operator, we want to introduce another important element of the R command line syntax: the assignment operator.

Probably without knowing about it, we've been using this already! When a dataset is imported into R Studio from the menu, a command is created such as

penelope21 <- read.csv("~/penelope21.csv")

What this does is look for the csv datafile called 'penelope21', and assign it to an object / variable called 'penelope21'

We could create any object name we wanted (within limits of names already known to R Studio). If we wanted to work with something called "week_3_data" we could use

week_3_data <- read.csv("~/penelope21.csv")

In R Studio, we would call on "week_3_data" rather than "penelope21" in the command lines.

4.2.4 Task 2 - Salary20 data

Using aggregate and summarise may not seem like much progress, because they are just replicating what we had already done with mean() is week 2. However

(a) this emphasizes that there are often several ways to get at the same thing in R (b) now we know about grouping, we can start to do more efficient and informative things.

Now, let's turn to the guesses made about median salary in the UK. We can get the data (created in 2020) from the file "salary20" in the week 3.zip file;

Let's take a peek at the dataset with

```
glimpse(salary20)
```

Glimpse pretty much does what you might think from the meaning of the word – it just gives us a data sample (handy because this is a much bigger dataset) and shows that we have 3 columns; UK_region (where someone lives, note 'other' probably equals Northern Ireland, Europe, China, etc, family_position (age relationship with siblings), and median_salary guess.

By the way, the govt statistics say the actual median income in 2019 was approx. £30,350 https://www.statista.com/statistics/1002964/average-full-time-annual-earnings-in-the-uk/

A) Can you use the "aggregate" command from task 1 to find out the salary guesses as a function of where someone lives? That is, can you adapt that code for this problem?
B) Can you use the aggregate command to find out salary guess as a function of family relationships? (if you are the youngest child maybe you have older siblings earning money that changes your evaluation?)
C) Can you get a breakdown of guess as a function of BOTH UK region AND family relationship together?
D) Can you adapt the "summarise" command to display salary guesses as a function of where someone lives?
E) Use R Studio to figure out the overall mean salary estimate and the standard deviation. Calculate by hand what salary estimate would have a z score of z=-1.5?

Chapter 5

Week 4: Z-score calculations and DVs/IVs again

Written by John Towse & Tom Beesley

5.1 Pre-lab work

This week - again, there's a learnr tutorial to follow and help prep for what we are covering: https://ma-rconnect.lancs.ac.uk/W4_LabPrep/

5.2 R Studio tasks

Last week we introduced two different ways to get a variable / column of scores investigated as a function of a separate column of data. In others words, describe the DV a function of an IV

Those two approaches involved

```
aggregate(x = DV, by = list(IV), FUN = mean)
```

and

```
data.frame %>% group_by(IV) %>% summarise(mean_estimate = mean(DV))
```

Use the week 3 learnr tutorial from last week to get more hints and practice with this, if you need to, alongside the week 4 tutorial.

5.2.1 Task 1 - Climate change data

1. Download the week4.zip file and upload it into R Studio server. If you need them, here are the key instructions from Week 2. This week the data is called "climate". Launch the week_4 R script as before.

Step 2. Once again, we're gong to be using commands from the tidyverse library (the pipe operator is one example) so we need to ensure that it's active. Run the command:

```
library(tidyverse)
```

(Nb. This command is the accompanying R script. You can access it from there.)

In the climate data frame, there are the following variables:

human_CC - A response to the question: How much do you think human activity contributes to climate change, as a percentage of overall climate change? Please enter a value between 0 (humans do not contribute to climate change at all) and 100 (human contribute entirely to climate change).

UK_region - where in the UK you used to live

country_visits - the number of countries you have visited

visit_category - a number between 1-3

- 1 means you have visited 1-4 countries
- 2 means you have visited 5-9 countries
- 3 means you have visited more than 9 countries

Step 3. Maybe belief in human activity as a contributor to climate change varies across the UK? Find out which region of the UK has the highest and the lowest belief in the relative role of human activity in climate change. Change the following command based on what was learnt last week and the current data described above

```
Data_frame_name %>% group_by(IV) %>% summarise(mean_estimate =
mean(DV))
```

Step 4. Maybe views about climate change are affected by how well travelled you are? Because you see the global impacts of clmate change, or because travel (often producing carbon emissions) is a sign of not being worried. So use the climate_change_category to look at whether there are differences in rating of human contribution to climate change.

5.2.2 Extra activity.

Come back to this afterwards for some extra practice if you want: Use the learnr activities to think about visualisation of the climate data. What graph(s) would be useful? How can you customise them?

5.3 Z-scores

> Hint / Reminder: Sketch a normal (z score) distribution and mark the mean/mode, and mark off the relevant parts of the question so you know what you are trying to achieve and how to interpret any calculations you make.
>
> Hint/ Reminder 2. For questions 6 & 7, remember that from the week 4 lecture material, typically in psychology we use the 5% level as a cutoff to decide, in broadly described terms, whether something is extreme or unlikely vs. at least somewhat plausible or likely.

5.3.1 z-scores 1: distributions

Q1. What is the relationship between the sign of a z-score and its position in a distribution?

Q2. If a distribution has a mean of 100 and a standard deviation of 10, what is the raw score equivalent to a z-score of 1.96?

Q3. If a distribution has a mean of 157 and a standard deviation of 19, what is the raw score equivalent to a z-score of 1?

5.3.2 z-scores 2: Using z-score tables

Q4. What proportion of scores lie between the mean and a z-score of 0.5?

Q5. What is the combined proportion of scores lying between z=-1.2 and z=.85?

5.3.3 z-scores 3: Applying z-scores to inferential problems

Q6. A Neuropsychologist has presented a test of face recognition to 200 neurotypical participants and finds that the scores are normally distributed with a mean of 85 and the standard deviation of 12. Two brain-damaged patients are also given the test. The one with right hemisphere brain damage scored 58 and the one with left hemisphere damage scored 67.

1. What is the z score of the right hemisphere patient when compared to the neurotypical group?
2. What proportion of neurotypical participants score lower that this patient?
3. Is this patient likely to belong to the population of neurotypical participants? (justify your answer)
4. What is the z score of the left hemisphere patient when compared to the neurotypical group?
5. What proportion of neurotypical participants score lower than this patient?
6. Is this patient likely to belong to the population of neurotypical participants? (justify your answer)

5.3.4 Extra activity

Come back to this afterwards for some extra practice if you want:

Q7. Tim Bizzley has measured the foot size of men and women and found each to be normally distributed. The men have a mean size of 55 with a standard deviation of 5 and the women a mean of 33 and a standard deviation of 5. Joanna Toes has foolishly measured two individuals but forgotten to note their gender. These have foot sizes of 37 and 47. To which gender is each more likely to belong? What evidence is there for this?

Chapter 6

Week 6: [NOT FOR CLASS TEST!] Visualising data and binomial tests

Written by Tom Beesley & John Towse

They say a picture paints a thousand words, so in this week's lab we will be learning some fundamental skills in **data visualisation** with the `ggplot()` commands. We will then conduct our first statisitical test to tell whether there is a *real difference* in our data. You must ensure you have watched the lecture and done the pre-lab work before coming to class.

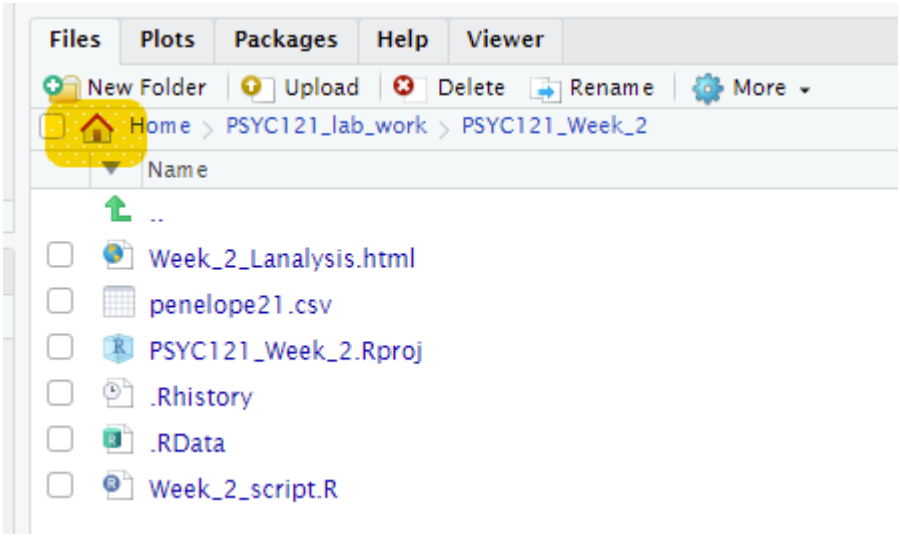
6.1 Pre-lab work: online tutorial

To access the pre-lab tutorial click here: https://ma-rconnect.lancs.ac.uk/Week6_LabPrep/ (on campus, or VPN required)

6.2 RStudio Task 1: Creating a new folder and project

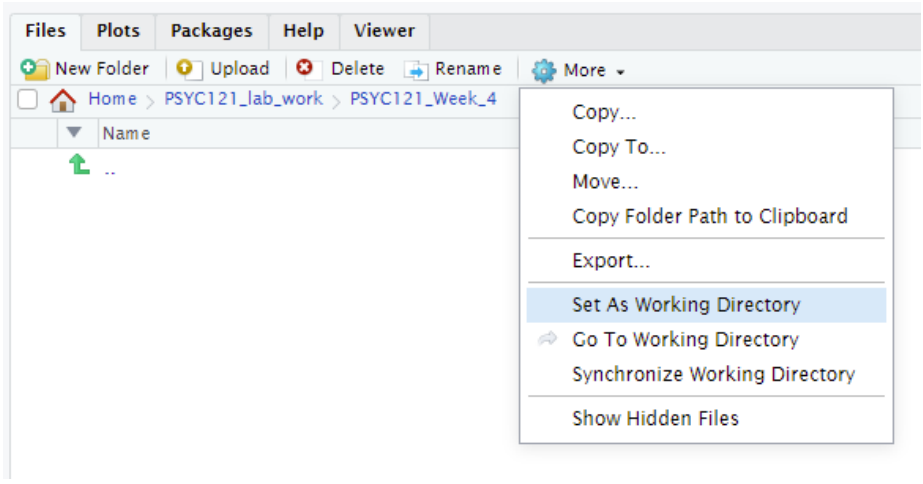
We are going to set up a new foler for this week and an RStudio *Project*. This is a good practice for organising your scripts and data.

1. When you are logged on to the RStudio Server, navigate to your home directory by clicking on the small house icon:



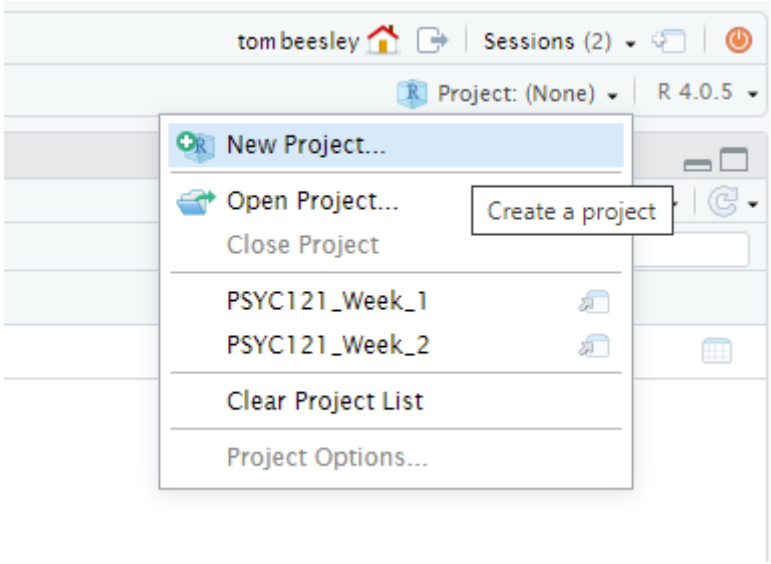
2. Click the "New Folder" button just above the home button. Name the folder something sensible (e.g., Week_6) and click OK.

3. Click on this folder and then click "More" and "Set As Working Directory". This tells RStudio that you are now working in this directory. When you read in data using `read.csv` it will know where to look for the file name that you provide.



4. Let's add the files we need for this week. Download the Week_6_files.zip and upload it into RStudio Server. If you need them, here are the key instructions from Week 2.

5. Now let's finish this process by making this a *Project*. Click on the small blue icon in the top right of RStudio, and clicking "New Project":



6. You may be asked to save the workspace or data - you should do this. Then select "Existing Directory" and make sure that the new directory you have created is selected as the "Project Working Directory" - it should be, if you set the working directory correctly above. If not, navigate to the correct directory here.

7. And you're done! This should now appear as a project in your front page on RStudio. You can get back to that front page by clicking the red power button in the top right corner. Using projects has many benefits. It will keep all the content for the week in one place, and save the commands you've used in the console. You can also use the *Project Menu* to navigate quickly between different projects.

6.3 RStudio Task 2: Visualising phone use

Now that you have completed the online tutorials on `ggplot()`, it's time to put those skills into practice with a new data set, provided in "data_phone.csv". In the survey, people estimated their daily phone use, and then looked up the actual time their phones were on.

1. Open the script "Week_6_script.R". This is the script file you will use for tasks 1 and 2

2. Run the first two lines of code `library(tidyverse)` and `data <- read_csv("data_phone.csv")`

3. You should now have an object in environment called `data`. You can view this (the "spreadsheet view") by clicking on the object.

4. Try running the ggplot command. You will see that this generates an error in red. Oh no!! Don't panic. Look at what it's telling you is wrong. Let's look at the ggplot command and edit it to make it display something. As a reminder from the online tutorial:

   > The main graphing command is `ggplot()` which tells R we want to draw some kind of plot. If you want to, you can just run `data %>% ggplot()`, and you'll see that R produces just a blank plot window. The plot is blank because we haven't told it a) what type of plot we want, and b) what data to plot.
   >
   > Note that the next line is **added** to `ggplot()` (i.e., `ggplot() + geom_point()`). Plotting with `ggplot()` involves adding elements (geoms) together. Think of it as adding layers on top of each other: the ggplot() is your pizza base, and the geoms and other features are the toppings!
   >
   > Within the `geom_point()` command we have the very important command of `aes()`. This stands for **aesthetics**. `aes()` tells ggplot how the data should be represented on the features of the plot.

5. Add y and x components inside the `aes()` command. These are used to map variables in your data to the features of the graph. If you are stuck on this, it's time to go back and look at the

6. Inside the `aes()` command, map the variable *screen_time_estimate* to the x axis, and the variable *screen_time_actual* to the y axis. In general, does it look like people's estimates were accurate?

7. Add a setting for *colour* outside of the `aes()`, to make all the points red. Why outside of `aes()`? Well remember from the online tutorial:

   > Careful observation of the code here will show you that these new commands are written **outside** of the `aes()` command. This is really important to note. We write these outside of `aes()` because these features of the plot **are not related to the data**. We are NOT *mapping* any of our data to these features. Instead, we are simply specifying one particular value for each of these "settings" in the plot

6. Now map the *colour* to the variable *phone_type* (within `aes()`). Can you see any differences between people who have different phones?

7. Try changing your `geom_point()` command to `geom_jitter()` (keeping the mappings the same). Suddenly this reveals many more points! Why is this? Check the help to find out more about this geom: `?geom_jitter`.

6.4 Task 3: conducting a sign test

Let's now look at whether these differences we see are meaningful. Do people estimate their daily screen time to be more than it actually is? There are two blocks of code to run here. You don't have to edit these, but it's important to look at what they are doing:

1. Run the first block of code. This adds a new column to the data, *est_diff*, which means the "estimate difference": actual screen time minus estimated screen time. You can take a look at this code by clicking on the new object in the environment, *data_phone_diff*. What do the positive and negative values reflect?

2. Run the second block of code. This filters out (!= means "not equal to") the zero values in the est_diff column. Why is this important for the sign test?

3. The result of the final line of code (`count()`) will be output in the console. This tells us how many values were above 0 (TRUE) and how many below zero (FALSE). This is all the information we need to conduct the sign test:

- How many observations were there in total?
- Use the sign test table below to determine how many values *against the majority* we would expect, according to the null hypothesis.
- Is the value in our data more or less than this?
- Does this mean there is a significant over or underestimate in people's estimates of phone time usage?

	level of significance for a one-tailed test						
	.10	.05	.025	.01	.005	.001	.0005
	level of significance for a two-tailed test						
N	.20	.10	.05	.02	.01	.002	.001
4	0						
5	0	0					
6	0	0	0				
7	1	0	0	0			
8	1	1	0	0	0		
9	2	1	1	0	0		
10	2	1	1	0	0	0	
11	2	2	1	1	0	0	0
12	3	2	2	1	1	0	0
13	3	3	2	1	1	0	0
14	4	3	2	2	1	1	0
15	4	3	3	2	2	1	1
16	4	4	3	2	2	1	1
17	5	4	4	3	2	1	1
18	5	5	4	3	3	2	1
19	6	5	4	4	3	2	2
20	6	5	5	4	3	2	2
21	7	6	5	4	4	3	2
22	7	6	5	5	4	3	3
23	7	7	6	5	4	3	3
24	8	7	6	5	5	4	3
25	8	7	7	6	5	4	4
26	9	8	7	6	6	4	4
27	9	8	7	7	6	5	4
28	10	9	8	7	6	5	5
29	10	9	8	7	7	5	5
30	10	10	9	8	7	6	5

Extra content!

Visualising data is all about communication: how can I communicate my results so that it makes the data very easy to interpret. Try these additional steps:

1. Add a `labs()` layer to modify text on the graph. If you're not sure how to do this, then feel free to go back to the online tutorial, or take a look at the help wiht `?labs`. Try changing the axis titles and add a title for the graph.
2. Change the theme of the graph. Try `?theme_classic` for help.
3. Graphs can be saved by clicking *Export*, then *Save as image*. Set the dimensions to how you like it to look, give it a suitable filename, and click save.