Statistics for Psychologists

John Towse, Tom Beesley

2022-10-21

Contents

Chapter 1

Week 1 - Introduction to PSYC121

Written by John Towse & Tom Beesley

1.1 Analysis labs and 'pre-lab' activity

For each lab session that you have been scheduled to attend, please come "prepared'. By this we mean

- You have watched the lecture video, made notes, and developed questions where you have them.
- You have worked through the online "prelab" preparation materials - Here is the one for Week 1
- You read the lab sheet (this one) to get a sense of what we're going to be doing, and you anticipate potential problems so that you can focus on these in the session.

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.

Want additional support? Keep in mind that the Department has endorsed and will use the Statistics textbook by David Howell called "**Fundamental statistics for the behavioral sciences"**

The book covers principles of statistics as well as some tutorials on using R. You can access a library copy from the library pages

1.2 Activities for this week

1.3 Task 1 - check-in with the University attendance register

When you arrive, make sure you have checked-in to your Analysis session in the Levy lab. All students are required by the University to confirm attendance at taught session

Staff will remind you of this in your class.

1.4 Task 2 - Getting dicy

Here's a simple task for you to complete as a group around each of the workstations;

You will be given a pair of dice

1. Working in pairs, one person rolls both dice.
2. Add up the total on each of them and have someone record that total (if you don't have some spare paper or a pen, use your computer)
3. Repeat those steps 20 times.
4. Then swap over your roles (the person rolling the dice, the person recording the outcome)

5. Once everyone at the workstation has had a turn at this, each person should attempt to work out (a) the mean and (b) the median of their dice roll total.
6. Check each others working, and discuss any differences or problems you have.

Are all your answers the same? Why / why not? If not, are they very different or very similar?

1.5 Task 3 - Using RStudio

1.5.1 Introducing R Studio

R and RStudio 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, RStudio is the interface for interacting with it. Put another way, *R is the engine, RStudio 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 make R your ally. Over the next year, in these labs we will increasingly be using RStudio 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.

1.5.2 Getting started with RStudio

For Lancaster University Psychology Students in 2022, 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 RStudio. 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. We're using the web server to make sure everyone has the same, controlled experience.

You will have received an email with your account information to log onto From a computer on the campus wifi, you can access R Studio at:

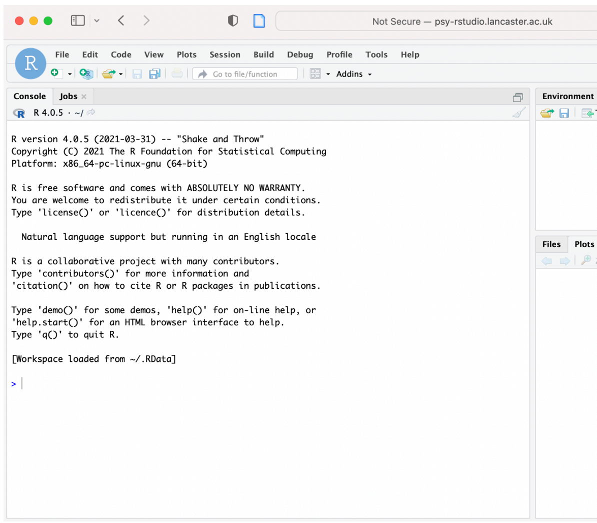
Lancaster Psychology R Studio Server

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

Your password for R Studio is: [password here]

As it says in the subject line, please keep your account details safe!

1.5.3 What does RStudio look like?



```
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

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).

1.5.4 Let’s get started!

The first thing we want to do in RStudio is to create a folder for this week so that we can put the relevant material there and keep it tidy.

From the lower-right panel of RStudio, click the files tab.

Select the “new folder” option and create a new folder (eg “week 1”)

Click on that folder to open it

Next, we’ve prepared some *instructions* for RStudio to use - this is called a “script”. So we need to get this script into the server for you to explore and play with

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
4. Click “Upload”
5. Click choose file and find the file on your computer.
6. 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 t.beesley@lancaster.ac.uk for support.

You should now have the script available in RStudio.

Use “Save…As” to create a new version of the script. By doing this, you’ll be able to have a “before” and “after” version of the script and can go back over the changes

In the script, select or highlight the first line of text, which is this one:

1.5.4.1 Run your first ever R instruction!

```r
5 + 5
```

and “run” this line. That tells RStudio to carry out the instruction.

You should see that in the console tab, RStudio calculates the answer to this incredibly hard maths challenge! (amazing huh? OK, maybe not *that* amazing…).

1.5.4.2 Modify your first ever R instruction!

Use your imagination – add a new line to the script and ask a different simple arithmetic question of your own choosing! What happens?

1.5.4.3 Calculate descriptive stats in R for the first time!

In this week's 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 used in the lecture. We've already created the instruction that will do exacly this and it is in the script, so run this line from the script

```
week_1_lecture_data <- c(7,8,8,7,3,1,6,9,3,8)
```

This creates an "object" called `week_1_lecture_data`. We can then perform calculations on this object. For example, we can find the mean by running the following command (use the script to do this)

```
mean(week_1_lecture_data)
```

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

Next, let's ask for the median by running this line from the script:

```
median(week_1_lecture_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 the block of code in the script for this that starts and ends with the "getmode" text (there are 6 lines of text)

This is just a bit of clever jiggery-pokery that gets the mode. What does R say the mode is?

1.5.4.4 Your challenge

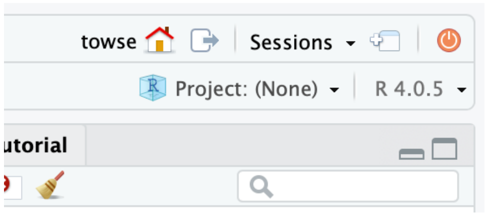
How can you get RStudio to verify / check the dice calculations that you attempted earlier? Think about how you might solve this problem, on the basis of what we have covered so far.

We will discuss this in class and attempt to get RStudio to check your answers. In doing so, annotate the script (add notes for you - not RStudio) using the "#" command

1.6 Before you finish

a) Make sure you save a copy of the script that you have been working on by the end of the session. This provides you with the record - the digital trace - on what you have done. And it means you can come back and repeat any of the work you have performed.

b) End your session on the RStudio server, this logs you out of the server and stops any ongoing activities and tasks you have set up, maybe in the background.



There is a red "power" button near the top right of the R studio window (do ask for help if you can't find it). It's a good habit to get into to turn the session off

1.6.1 Extra content for outside the lab class

a) In the Howell text book on statistics, there's some R code on descriptive statistics. It is included in the script for you to look at and play with.

b) 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 numbers 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 ifyou-didn'thaveanyspaces)

1.7 Task 4 – Review the learnr sample / practice questions

After every block of teaching in part-1 analysis (specifically, we mean in week 5, week 10, week 15 and week 20) there will be a class test. This will assess your knowledge and your understanding of the material that has been covered.

The class test will comprise a set of Multiple Choice Questions (and the set of questions will be different for each student, as the test will involve random selection from a larger pool) under timed conditions.

In order to help you get (a) a broad or basic feel for the sort of questions you might get in the class test (b) self-review your progress through the term, we will provide MCQs each week for you to attempt.

So these are for your benefit… you can take the questions when you choose to, and the learnr quiz will provide feedback on the answers your provide. Just bear in mind:

a) We place a set of questions at the end of the learnr pages so that you can attempt these at the end of each week, after you've completed lab activities, follow-up work, weekly Q&As etc. But it's up to you when you answer the questions

b) These are meant as *indicative questions*. There's no point in learning/ memorising these questions (they won't be on the quiz!) and our advice is to reflect on how the teaching and content links to the sorts of questions that get posed.

1.8 Task 5 – Data collection exercise

In order to learn about psychology and data analysis techniques, we need data! Rather than rely too much on artificial data (certainly it is sometimes useful to say "Here are a bunch of numbers and this is what we can do with them" – think about the R Studio example for this week's lab) for the most part, we prefer to draw on datasets that are a bit more engaging and meaningful that you have a stake in yourself! By using a common data set, that we can return to over the year, we can also build up familiarity and confidence in the data and remove a potential obstacle to thinking about the more important analysis part.

So a key task will be for everyone to have a go at taking our online survey, and contribute to a dataset that can be used throughout the year.

We would like you to complete the survey via your Department Sona system account. This way, you will receive a research credit for doing so, to "jump start" your credit account.

The sona system is can be accessed by following this link

Chapter 2

Week 2: Descriptive statistics in RStudio

2.1 Pre-lab work

Last week, among other things, we asked you to

- Roll some dice, carry out some (relatively) straightforward ‘hand’ calculations of central tendency
- Connect to the RStudio server and create a folder
- Within RStudio upload and run a script - a set of instructions, and explore annotations
- Adapt the script commands to perform calculations on the dice rolls within RStudio
- Complete a survey so that we can collect data for analysis teaching

Your progress was great! We start with small steps and build up - but this is a nice start and we’re pleased how things went!

Before the lab, make sure you have worked through the material in the week2 learnr tutorial. The link is here

2.1.1 R Studio tasks

For a reminder of how to start RStudio, see week 1 lab sheet

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

> A word of advice (from David Howell's statistics book: "One more word of advice"I can't resist adding what is perhaps the best advice I have. If there is something that you don't understand, just remember that "Google is your friend." She certainly is mine. (Well, maybe Google is getting a bit pushy, but there are many other search sites.) If you don't understand what Fisher's Exact Test is, or you don't like my explanation, go to Google and type in Fisher's Exact Test. I just did that and had 260,000 hits. You can't tell me that there isn't going to be something useful in there.)"

2.1.2 Bringing data, scripts and files into R Studio

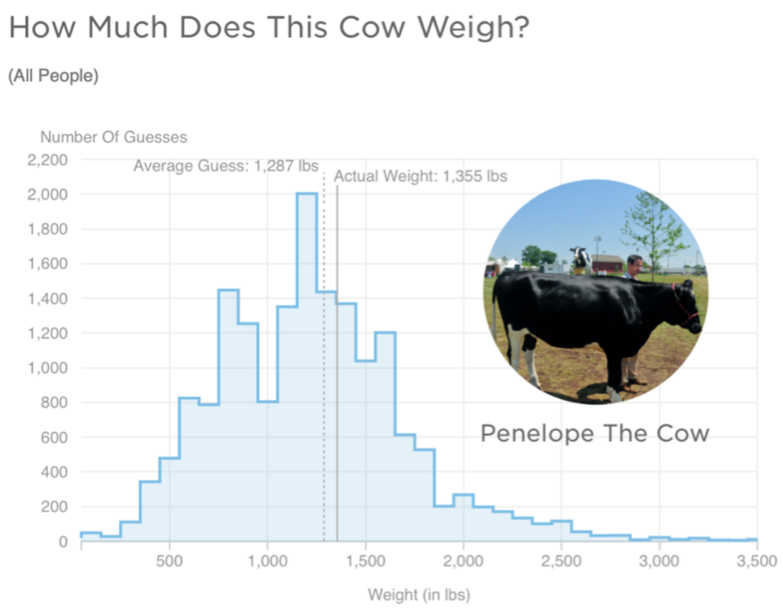
In week 1, we had a tiny dataset (relatively speaking) that we entered into R through a script line. That worked for what is was. But it's going to become painful and tedious when (a) we want to work with larger datasets (b) we have data more complex than a 1-dimensional list of numbers (think about some 2-dimensional data sheets you might have encountered in excel for example)

R can handle data files, and this week we're going to explore them. Within R, we can specify 'data frames' which can have, essentially, multiple columns of data, and we can link data files to data frames for processing

To make things straightforward, each week we'll provide students with a "zip" file that contains the script to start from (which you can expand and annotate etc, and save on your file space). We'll also provide a data file or data files for you to use in the zip file. R can then *import* these files into the RStudio environment. So when you upload the zip file, you can import the data AND you can open up the script

2.2 Lab exercises - descriptive information in R Studio

Some 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 the 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 sample of) the PSYC121 student data collected on guessing the weight of Penelope, and ask whether it resembles the properties of this large dataset.

2.2.1 Loading the data

Using the instructions and advice given on Moodle, get the `week2.zip` file and bring the data and R script into R Studio. The week 2 zip file is here

2.2.2 Using the R script

Step 1. Download the `week_2.zip` file (If you are using a mac, there is a video guide on Moodle to explain how to download the zip file as is)

Step 2. Open the appropriate folder on the RStudio server

Step 3. Upload the zip file

Let's start working with our data, by opening up (clicking on) the script "Week_2.R" file.

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)

The data are on the RStudio server 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

```
cows <- read_csv("~/penelope22.csv")
```

What this command accomplished was to read the spreadsheet called 'penelope22' into an object in R called `cows`. 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(cows)
```

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

2.2.3 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(week_1_lecture_dataa)
```

That gave us the mean of the small dataset `week_1_lecture_data`. This time, we want to explore the penelope dataset. But also, the lecture_data was just a single list of numbers. The penelope22 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**. So run the followinbg line in the script

```
mean(cows$estimate)
```

```
sd(cows$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?

2.2.4 Calculations from a range of columns

We have seen that:

```
mean(cows$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 change the script so it looks like this:

```
mean(cows$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)

2.2.5 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):

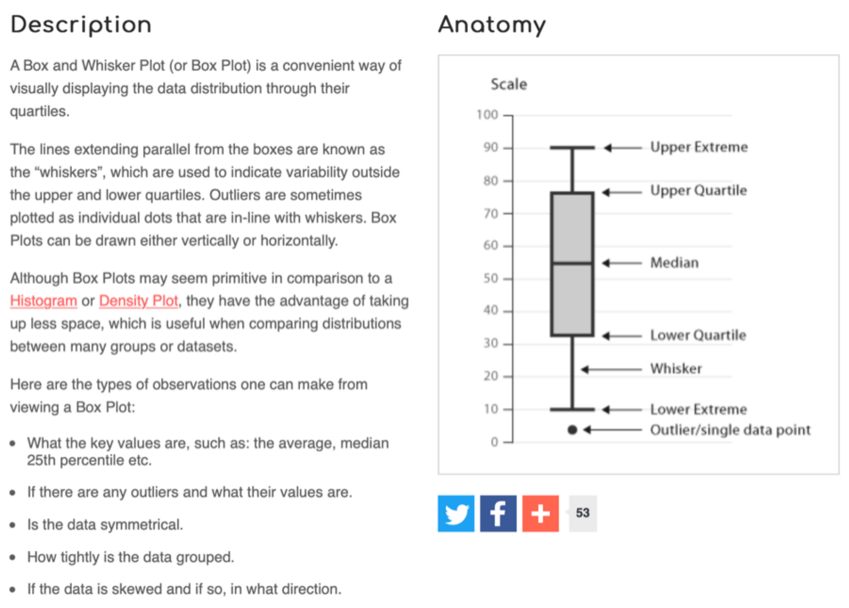
```
hist(cows$estimate)
```

One way to alter or adjust the histogram is to change the width of the bars, the intervals, between each plot section. Try run this line from the script

```
hist(cows$estimate, breaks = ??)
```

Does it work? No? What you need to do is replace the two question marks in the script (or better still, create a new instruction line in which you amend this to have a numerical value representing the number of different plot bars. Try at least 3 different values. Look at and think about how this affects the visual distribution.

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:



Description

A Box and Whisker Plot (or Box Plot) is a convenient way of visually displaying the data distribution through their quartiles.

The lines extending parallel from the boxes are known as the "whiskers", which are used to indicate variability outside the upper and lower quartiles. Outliers are sometimes plotted as individual dots that are in-line with whiskers. Box Plots can be drawn either vertically or horizontally.

Although Box Plots may seem primitive in comparison to a Histogram or Density Plot, they have the advantage of taking up less space, which is useful when comparing distributions between many groups or datasets.

Here are the types of observations one can make from viewing a Box Plot:

- What the key values are, such as: the average, median 25th percentile etc.
- If there are any outliers and what their values are.
- Is the data symmetrical.
- How tightly is the data grouped.
- If the data is skewed and if so, in what direction.

Anatomy

2.2.6 Additional challenge

In the zip file, we also provide data on the estimates of the percentage of immigrants in the UK. This will allow you to explore the data discussed in the analysis lecture, and create visualisations of the data and its spread. Can you apply the analysis of the penelope data to the immigration data?

Chapter 3

Week 3: DVs and IVs in RStudio

Written by John Towse & Tom Beesley

3.1 Pre-lab work

Last week we asked you to

- Use a script to run instructions in RStudio
- Put data into RStudio form a data file and explore how to run descriptive stats

This week - again, there's a learnr tutorial to follow and help prep for this week's activities/ You can find it here

3.2 R Studio tasks

For the "penelope22" 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...

3.2.1 Task 1 - More with the penelope22 data

Step 1. This week, we again 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)
```

Step 2. Explore help() commands. R can give you more information about how it works.

Step 3. Creating a project and a folder, and setting the working directory

Step 4. Bring the `week3.zip` file into R Studio server. Like last week, upload the zip file, and launch the R script.You can get the file here

Step 5. Read or load the penelope data into R. Have a look at it using the View() command in the script

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

```
aggregate(x = *MISSING*2$estimate, by = list(*MISSING*$identity),
FUN = mean)
```

First, let's try this (you will need to use your object name in place of MISSINFG! What do you get? Does this match what we did last week when we calculated the mean for the *female* and for the *other* (i.e., non-female) group?

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

`aggregate` This is a command to get descriptive statistics

`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?)

3.2.2 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 called 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)

3.2.3 The assignment operator

As well as learning about the pipe operator, we want to remind you /draw attention explicitly to another important element of the R command line syntax: the assignment operator.When a dataset is imported into RStudio from the menu, a command is created such as

```
cows <- read_csv("penelope22.csv")
```

What this does is look for the csv datafile called 'penelope22', and assign it to an object / variable called 'cows'

We could create any object name we wanted (within limits of names already known to RStudio). It isn't just for reading in data files, we can perform a whole range of functions and assign them to an object.

3.2.4 Task 2 - Salary 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, about working with 2-dimensional data, 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 from the file "wages" in the week_3.zip file (you will need to adapt the code in the script for the penelope data so that it will load in the wages data)

Let's take a peek at the dataset with

```
glimpse(wages)
```

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 salary (estimate).

By the way, the govt statistics say the actual median income in 2019 was approx. £30,350 see this link

A) Writing into your script, use the working “aggregate” commands from task 1 with the penelope weight data to find out the salary guesses as a function of where someone lives? That is, can you adapt that code for this problem? First, make sure you read in the data file into an R object.

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 use the `group by()` command to display salary guesses as a function of where someone lives? Check this this gives you the same answer

3.2.5 Task 3 - Phone Use data

E) Datset 3: Use the dataset of phone screen time usage to further explore and consolidate the group_by() command (ie we’ll drop the aggregate command for this task to focus on group_by(). Adapt script lines from the above two tasks to read in and calculate screen time usage as a function of the type of phone.

F) Use RStudio 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?

3.2.6 Task 4 - Final challenge - visuallisation

Can you find a way to visualise the screentime usage data that you have been working with above? The script provides two ways to consider doing this - boxplots (which we have looked at in script commands already) and ggplot, which we have spent less time with but is an extremely powerful engine for creating plots. We’ve provided the start of the code in each case, leaving you to work out the specifics. Remember to annotate your creations!

3.2.7 Now you’re finished ...

Remember to complete your notes in the script, save the script file, and end the session.