Analysing and Interpreting Psychological Data II - PSYC402

Margriet Groen and Rob Davies

2022-01-25

Contents

Chapter 1

Intro

This is a collection of tuition material written for Psychology postgraduates at Lancaster University. At the moment the content represents the "lab materials" for the PSYC402 module. They feature tuition of programming with R, building on the skills you developed last term.

1.1 Analysis labs and 'pre-lab work'

Some parts should be completed before you attend the lab session (watching lectures, reading chapters, pre-lab activities). All the links to the different materials and activities are also in the 'to-do list' for the relevant week on Moodle.

Chapter 2

Week 11: Recap of the linear model and practising data-wrangling in R

Written by Margriet Groen (partly adapted from materials developed by the PsyTeachR team a the University of Glasgow)

Before we start covering new material, we want to spent some time on recapping the basic concepts of the linear model (correlation, simple regression, multiple regression). You all come from different educational backgrounds and therefore have vastly different knowledge of, and experience with statistics. Therefore, please follow your own judgement as to whether you feel you want to/need to revisit material outlining the theoretical background to and the practical implementation in R for these topics. Below we provide some guidance as to materials that are relevant. **Just to be clear: We don't expect you to watch and/or read and/or do everything, please have a look at what you feel you need and spend some time with those materials.**

2.1 Lectures

The linear model was discussed in weeks 6 to 9 of PSYC401, so that is a good place to start.

Alternatively, if you don't feel confident about the material, these recorded lectures might help.

- **The linear model: theory (~30 min)** An introduction to the linear model and linear regression. I follow material as discussed in Chapter 4

of Bodo Winter's book *Statistics for Linguists: An Introduction using R* (see below under 'Reading').

- **How to build a linear model in R (~30 min)** In this video I demonstrate how to build a linear model in R by talking you through a simple linear regression script (you can download it here stats_linearModel_howTo.R). If you are unclear on what different parts of the lm() function do, or how to read the output, this video might help clarify that.
- **Multiple regression: theory (~35 min)** An introduction to multiple regression. I follow material as discussed in Chapter 5 of Bodo Winter's book *Statistics for Linguists: An Introduction using R* (see below under 'Reading').
- **Centering and standardising (~5 min)** Brief explanation of what centering and standardising are.

2.2 Reading

2.2.1 Miller & Haden (2013)

Link

Chapter 10 gives you a brief overview of what correlation and regression are. **Chapter 11** introduces correlation in more detail. **Chapters 12 and 14** provide accessible overviews of simple and multiple regression, respectively. All these chapters are really short but provide a good basis to understanding. We consider this the minimum level of understanding you should acquire.

2.2.2 Winter (2020)

Link

Chapter 4 provides and excellent conceptual introduction to the linear model and also explains how this is implemented in R (highly recommended).

Chapter 5 takes a slightly different approach to the one taken in Miller & Haden (2013) to introducing correlation. If you already understand the basic theory behind correlation, this will be an interesting read. Chapter 5 also clearly explains what centering and standardizing are and why you need to bother with these linear transformations.

Chapter 6 provides an excellent overview of multiple regression and also explains how this is implemented in R.

2.3 Pre-lab activities

After having watched the lectures and read the textbook chapters you'll be in a good position to try these activities. Completing them before you attend your lab session will help you to consolidate your learning and help move through the lab activities more smoothly.

2.3.1 Pre-lab activity 1: Visualising the regression line

Have a look at **this visualisation of the regression line** by Ryan Safner.

In this shiny app, you see a randomly-generated set of data points (within specific parameters, to keep the graph scaled properly). You can choose a slope and intercept for the regression line by using the sliders. The graph also displays the residuals as dashed red lines. Moving the slope or the intercept too much causes the generated line to create much larger residuals. The shiny app also calculates the sum of squared errors (SSE) and the standard error of the regression (SER), which calculates the average size of the error (the red numbers). These numbers reflect how well the regression line fits the data, but you don't need to worry about those for now.

In the app he uses the equation Y = aX + b in which b is the intercept and a is the slope.

This is slightly different from the equation you saw during the lecture. There we talked about Y = b0 + b1*X + e. Same equation, just different letters. So b0 in the lecture is equivalent to b in the app and b1 in the lecture is equivalent to a in the app.

Pre-lab activity questions:

1. Change the slider for the intercept. How does it change the regression line?
2. Change the slider for the slope. How does it change the regression line?
3. What happens to the residuals (the red dashed lines) when you change the slope and the intercept of the regression line?

2.3.2 Pre-lab activity 2: Data-wrangling in R

In PSYC401, you've already learned how to read in data, how to select variables and how to compute summary statistics, so re-visiting the PSYC401 materials is a good place to start.

RStudio also provides some useful interactive tutorials that take you through the basics:

- **The Basics** Start here to learn how to inspect, visualize, subset and transform your data, as well as how to run code.
- **Work with Data** Learn how to extract values form a table, subset tables, calculate summary statistics, and derive new variables.
- **Visualize Data** Learn how to use ggplot2 to make any type of plot with your data. The tutorials on **Exploratory Data Analysis** and **Scatterplots** are particularly relevant.

Please note that there are often different ways to do the same or similar things in R. This means you might encounter slightly different functions or styles of coding in different materials. This is not something to worry about. Just make sure you're clear on what a bit of code achieves and choose the function/style that you feel most comfortable with.

2.3.3 Pre-lab activity 3: Getting ready for the lab class

2.3.3.1 Remind yourself of how to access and work with the RStudio Server.

- **Video on how to access the RStudio Server by Padraic**
- I highly recommend using R Projects to structure your workflow. You could create an R project for each week of the module. Have a look at section **8 Workflow: projects** of **R for Data Science** by Hadley Wickam and Gareth Grolemund for an introduction.

2.3.3.2 Get your files ready

Download the 402_week11_forStudents.zip file and upload it into a new folder in RStudio Server.

2.4 Lab activities

In this lab, you'll gain understanding of and practice with:

- when and why to apply simple and multiple regression to answer questions in psychological science
- conducting multiple regression in R
- interpreting the R output of simple and multiple linear regression
- reporting results for simple and multiple linear regression following APA guidelines

2.4.1 Lab activity 1: Interpreting and reporting results

Have a look at the R output below.

R Output 1

```
Call:
lm(formula = wordReading ~ nonWordReading, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-0.40929 -0.12762 -0.02772  0.13377  0.50954

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)      1.2908     0.1294   9.977 1.74e-13 ***
nonWordReading   0.7602     0.0915   8.308 5.57e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1938 on 50 degrees of freedom
Multiple R-squared:  0.5799,    Adjusted R-squared:  0.5715
F-statistic: 69.03 on 1 and 50 DF,  p-value: 5.568e-11
```

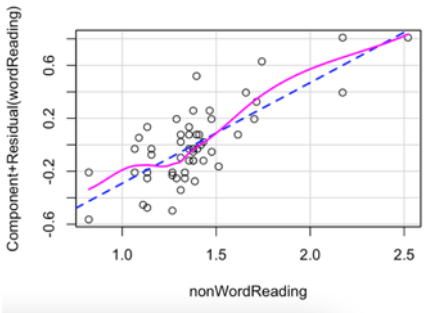
1. What is the outcome or dependent variable?
2. What is the predictor or independent variable?
3. Is the overall model significant?
4. How much variance does the model account for?

Thinking about assumptions, what do you conlcude from the plots and output below?

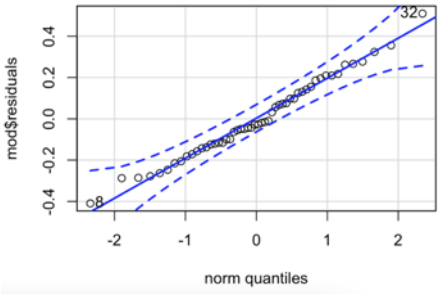
5. Does the relationship appear linear?

6. Do the residuals show normality and homoscedasticity?

Scatterplot



QQ-plot



R Output 2

```
        Shapiro-Wilk normality test

data:  mod$residuals
W = 0.98669, p-value = 0.8254

Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.1392492, Df = 1, p = 0.70903
```

2.4.2 Lab activity 2: Conducting simple and multiple regression

2.4.2.1 Background

Today, to help get a practical understanding of regression, you will be working with real data and using regression to explore the question of whether there is a relationship between voice acoustics and ratings of perceived trustworthiness.

2.4.2.1.1 The Voice The prominent theory of voice production is the source-filter theory (Fant, 1960) which suggests that vocalisation is a two-step process: air is pushed through the larynx (vocal chords) creating a vibration, i.e. the source, and this is then shaped and moulded into words and utterances as it passes through the neck, mouth and nose, and depending on the shape of

those structures at any given time you produce different sounds, i.e. the filter. One common measure of the source is pitch (otherwise called Fundamental Frequency or F0 (F-zero)) (Titze, 1994), which is a measure of the vibration of the vocal chords, in Hertz (Hz); males have on average a lower pitch than females for example. Likewise, one measure of the filter is called formant dispersion (measured again in Hz), and is effectively a measure of the length of someone's vocal tract (or neck). Height and neck length are suggested to be negatively correlated with formant dispersion, so tall people tend to have smaller formant dispersion. So all in, the sound of your voice is thought to give some indication of what you look like.

More recently, work has focussed on what the sound of your voice suggests about your personality. McAleer, Todorov and Belin (2014) suggested that vocal acoustics give a perception of your trustworthiness and dominance to others, regardless of whether or not it is accurate. One extension of this is that trust may be driven by malleable aspects of your voice (e.g. your pitch) but not so much by static aspects of your voice (e.g. your formant dispersion). Pitch is considered malleable because you can control the air being pushed through your vocal chords (though you have no conscious control of your vocal chords), whereas dispersion may be controlled by the structure of your throat which is much more rigid due to muscle, bone, and other things that keep your head attached. This idea of certain traits being driven by malleable features and others by static features was previously suggested by Oosterhof and Todorov (2008) and has been tested with some validation by Rezlescu, Penton, Walsh, Tsujimura, Scott and Banissy (2015).

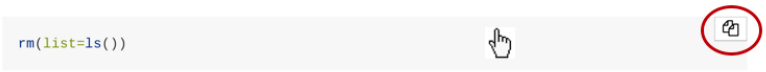
So, the research question today is: Can vocal acoustics, namely pitch and formant dispersion, predict perceived trustworthiness from a person's voice? We will only look at male voices today, but you have the data for female voices as well should you wish to practice (note that in the field, tendency is to analyse male and female voices separately as they are effectively sexually dimorphic). As such, we hypothesise that a linear combination of pitch and dispersion will predict perceived vocal trustworthiness in male voices. This is what we will analyse.

To complete this lab activity, you can use the R-script (402_wk11_labAct2.R) that you downloaded as part of the 'pre-lab activities' as a template. Work through the activity below, adding relevant bits of code to your script as you go along.

2.4.2.2 Step 1: Background and set up

Before you do anything else, when starting a new analysis, it is a good idea to empty the R environment. This prevents objects and variables from previous analyses interfering with the current one.

> **TASK**: Use the code snippet below to clear the environment. **TIP**: If you hover your mouse over the box that includes the code snippet, a 'copy to clipboard' icon will appear in the top right corner of the box. Click that to copy the code. Now you can easily paste it into your script.




```
rm(list=ls())
```

Before we can get started we need to tell R which libraries to use. For this analysis we'll need broom, car, pwr and tidyverse.

> **TASK**: Load the relevant libraries. **HINT**: Use the `library()` function.

In this lab, we are setting out to test whether a linear combination of pitch and dispersion will predict perceived vocal trustworthiness in male voices. We'll be working with two data files:

- voice_acoustics.csv - shows the VoiceID, the sex of the voice, and the pitch and dispersion values
- voice_ratings.csv - shows the VoiceID and the ratings of each voice by 28 participants on a scale of 1 to 9 where 9 was extremely trustworthy and 1 was extremely untrustworthy.

> **TASK**: Read in both files (using the `read_csv()` function), have a look at the layout of the data and familiarise yourself with it. The ratings data is rather messy and in a different layout to the acoustics but can you tell what is what?

QUESTION 1 How are the acoustics data and the ratings data organised (wide or long)? Are both data files 'tidy'? If you need more info on what that means, have a look **here**.

2.4.2.3 Step 2: Restructuring the ratings data

We are going to need to do some data-wrangling before we do any analysis! Specifically, we need the change the ratings data to the long format.

Here we'll use the `pivot_longer()` function (see **here** or type `?pivot_longer` in the Console for more info) to restructure the ratings data from wide to long and store the resulting table as 'ratings_tidy'.

> **TASK**: Use the code snippet below to restructure the data.

```
ratings_tidy <- pivot_longer(
  data = ratings,     # the data you want to restructure
  cols = P1:P28,      # columns you want to restructure
```

```
  names_to = "participant", # variable name that captures whatever is across the columns
  # (in this case P1 to P28 for the 28 different participants)
  values_to = "rating") # variable name that captures whatever is in the cells
  # (in this case numbers for ratings)
```

2.4.2.4 Step 3: Calculate mean trustworthiness rating for each voice

Now that we have the ratings data into a tidy format, the next step is to calculate the mean rating for each voice. Remember that each voice is identified by the 'VoiceID' variable.

> **TASK**: Calculate the mean rating for each voice and store the resulting table in a variable named 'ratings_mean'. **HINT**: Use `group_by()` and `summarise()`. Are you using the tidy data? Also, remember that if there are any missing values (NAs) then na.rm = TRUE would help.

2.4.2.5 Step 4: Join the data together

Ok, before we get ahead of ourselves, in order to perform the regression analysis we need to combine the data from 'ratings_mean' (the mean ratings) with 'acoustics' (the pitch and dispersion ratings). Also, as we said, we only want to analyse male voices today.

> **TASK**: Join the two tables and keep only the data for the male voices, call the resulting table 'joined'. **HINT**: Use the `inner_join()` function (making use of the variable that is common across both tables) to join. See **here** or type `?inner_join` in the Console for more info. Use the `filter()` function to only keep male voices. Remember that the Boolean operator for exactly equal is `==`.

2.4.2.6 Step 5: Spreading the data

Ok so we are starting to get an understanding of our data and we want to start thinking about the regression. However, the regression would be easier to work with if Pitch and Dispersion were in separate columns. This can be achieved using the `pivot_wider()` function (see **here** or type `?pivot_wider` in the Console for more info). This is basically the inverse of `pivot_longer()`. It increases the number of columns and decreases the number of rows.

> **TASK**: Use the code snippet below to spread the data.

```
joined_wide <- joined %>%
  pivot_wider(
    names_from = measures, # name of the categorical column to spread
    values_from = value) # name of the data to spread
```

QUESTION 2 Why do we not need to specify within the `pivot_wider()` function which data to use?

2.4.2.7 Step 6: Visualising the data

As always, it is a good idea to visualise your data.

> **TASK**: Now that we have all the variables in one place, make two scatterplots, one of mean trustworthiness rating with dispersion and one for mean trustworthiness rating and pitch. **HINT**: For this you'll need the `ggplot()` function together with `geom_point()` and `geom_smooth()`. Make sure to give your axes some sensible labels.

QUESTION 3 According to the scatterplots, how would you decribe the relationships between trustworthiness and dispersion and trustworthiness and pitch in terms of direction and strength? Which one of the two seems stronger?

2.4.2.8 Step 7: Conducting and interpreting simple regression

With all the variables in place, we're ready now to start building two simple linear regression models:

1. Predicting trustworthiness mean ratings from Pitch
2. Predicting trustworthiness mean ratings from Dispersion

> **TASK**: Use the `lm()` function to run the following two regression models and use the `summary()` function to look at the output of each model. Store the first model in a table called 'mod_pitch' and store the second model in 'mod_disp'. **HINT**: `lm(dv ~ iv, data = my_data)`

QUESTION 4 What do you conclude from the output of these models? Which model is significant? Which predictors are significant? How much variance does each model describe?

2.4.2.9 Step 8: Conducting and interpreting multiple regression

Now let's look at both predictors in the same model. Before we do this, it is sensible to center and standardise the predictors.

Look at the code below. Can you follow how the predictors are first centered (_c) and then standardised (_z)?

Here I do this by hand because I think it makes it clearer, even though there are functions that do this in one step (`scale()`).

```
joined_wide <- mutate(joined_wide,
                      Dispersion_c = Dispersion - mean(Dispersion),
                      Dispersion_z = Dispersion_c / sd(Dispersion_c),
                      Pitch_c = Pitch - mean(Pitch),
                      Pitch_z = Pitch_c / sd(Pitch_c))
```

> **TASK**: Now use the centered and standardised data for the multiple regression. Use the `lm()` function to run a model for predicting trustworthiness mean ratings from Pitch and Dispersion, and store the model in 'mod_pitchdisp_z'. Use the 'summary()' function to look at the output. **HINT**: `lm(dv ~ iv1 + iv2, data = my_data)`

QUESTION 5 What do you conclude from the output of this model? Is the overall model significant? Which predictors are significant? How much variance does the model describe? Which model would you say is best for predicting ratings of trustworthiness, the Pitch only, the Dispersion only or the Pitch+Dispersion model?

2.4.2.10 Step 9: Checking assumptions

Now that we've established which model best fits the data, let's check whether it meets the assumptions of linearity, normality and homoscedasticity.

> **TASK**: Check the assumptions of linearity, normality and homoscedasticity. **HINT**: `crPlots()` to check linearity `qqPlot()` and `shapiro.test()` to check normality of the residuals `residualPlot()` and `nvcTest()` to check homoscedasticity of the residuals

QUESTION 6 What do you conclude from the graphs and output? Should we also check for collinearity?

2.4.2.11 Step 10: Writing up the results

> **TASK**: Write up the results following APA guidelines. **HINT**: **The Purdue writing lab website** is helpful for guidance on punctuating statistics. The **APA Style 7th Edition Numbers and Statistics Guide** is also useful.

2.5 Answers

When you have completed all of the lab content, you may want to check your answers with our completed version of the script for this week. **Remember**, looking at this script (studying/revising it) does not replace the process of working through the lab activities, trying them out for yourself, getting stuck, asking questions, finding solutions, adding your own comments, etc. **Actively engaging** with the material is the way to learn these analysis skills, not by looking at someone else's completed code…

2.5.1 Lab activity 1: Interpreting and reporting results

1. What is the outcome or dependent variable? **Word reading**
2. What is the predictor or independent variable? **Non-word reading**
3. Is the overall model significant? **Yes, $F(1,50) = 69.03$, $p < .001$**
4. How much variance does the model account for? **58%**
5. Does the relationship appear linear? **Yes. The dots and the pink line assemble quite closely on the dashed line.**
6. Do the residuals show normality and homoscedasticity? **The qq-plot suggest that the residuals are normally distributed as the dots fall close to the solid blue line and within the range of the dashed blue lines. The Shapiro-Wilk test of normality confirms this (it is not significant). Similarly, the output of the non-constant variance score tests is not significant suggesting that the residuals are homoscedastic.**

2.5.2 Lab activity 2: Conducting simple and multiple regression

You can download the R-script that includes the relevant code and answers to the questions here: 402_wk11_labAct2_withAnswers.R.

Chapter 3

Week 12: Categorical predictors

Written by Margriet Groen (partly adapted from materials developed by the PsyTeachR team a the University of Glasgow)

So far, all the predictors in the models we've looked at were continuous variables. What if you wanted to know whether a response differed between two or more discrete groups? Hang on, you might say, that sounds like doing an ANOVA. True, you might have used ANOVA to assess whether group means differed in previous stats courses. ANOVAs—to some degree—are just a special type of regression where you have categorical predictors. This week we'll look at how to model responses as a function of categorical predictors and we'll combine categorical predictors to model how a predictor might affect the outcome variable differently across two different groups. For example, we might be interested in whether the amount of time adolescents use digital devices (screen-time) predicts their well-being. Additionally, we might want to know whether well-being is different for adolescent boys and girls and whether the relationship between screen-time and well-being differs for these two groups. By fitting a regression model in which we combine a continuous (screen-time) and a categorical (sex) predictor, we can do exactly that. We'll be working on that in the lab.

3.1 Lectures

The lecture material for this week follows the recommended chapters in Winter (2020) – see under 'Reading' below – and is presented in two parts:

- **Categorical predictors (~17 min)**
- **Interactions (~18 min)**

3.2 Reading

3.2.1 Blogpost by Professor Dorothy Bishop

In this **very short blogpost** Professor Dorothy Bishop explains the links between ANOVA and Regression.

3.2.2 Winter (2020)

Link

Chapter 7 provides an excellent overview of using categorical predictors in regression models and explains how this is implemented in R.

Chapter 8 explains what interactions are and how to model and interpret them.

3.3 Pre-lab activities

After having watched the lectures and read the textbook chapters you'll be in a good position to try these activities. Completing them before you attend your lab session will help you to consolidate your learning and help move through the lab activities more smoothly.

3.3.1 Pre-lab activity 1: Data-wrangling in R

The more you practise coding in R, the easier it will become. The RStudio interactive tutorials I mentioned last week are an excellent place to start if you haven't engaged with those yet.

- **The Basics** Start here to learn how to inspect, visualize, subset and transform your data, as well as how to run code.
- **Work with Data** Learn how to extract values form a table, subset tables, calculate summary statistics, and derive new variables.
- **Visualize Data** Learn how to use ggplot2 to make any type of plot with your data. The tutorials on **Exploratory Data Analysis** and **Scatterplots** are particularly relevant.

If you feel confident with the material covered in those tutorials the following are useful to try:

- **Separating and Uniting Columns** Here you will learn to separate a column into multiple columns and to reverse the process by uniting multiple

columns into a single column. Then you'll practise your data wrangling skills on messy real world data.

- **Join Data Sets** Learn how to work with relational data. Here you will learn how to augment data sets with information from related data sets, as well as how to filter one data set against another.

Please note that there are often different ways to do the same or similar things in R. This means you might encounter slightly different functions or styles of coding in different materials. This is not something to worry about. Just make sure you're clear on what a bit of code achieves and choose the function/style that you feel most comfortable with.

3.3.2 Pre-lab activity 2: Getting ready for the lab class

3.3.2.1 Remind yourself of how to access and work with the RStudio Server.

- **Video on how to access the RStudio Server by Padraic**
- I highly recommend using R Projects to structure your workflow. You could create an R project for each week of the module. Have a look at section **8 Workflow: projects** of **R for Data Science** by Hadley Wickam and Gareth Grolemund for an introduction.

3.3.2.2 Get your files ready

Download the 402_week12_forStudents.zip file and upload it into a new folder in RStudio Server.

3.4 Lab activities

In this lab, you'll gain understanding of and practice with:

- when and why to apply multiple regression to answer questions in psychological science
- conducting multiple regression in R when combining continuous and categorical predictors
- interpreting the R output of multiple linear regression (when combining continuous and categorical predictors)
- reporting results for multiple linear regression (when combining continuous and categorical predictors), following APA guidelines

3.4.1 Lab activity 1: Combining a continuous and a categorical predictor in a resssion model

3.4.1.1 Background: Smartphone screen-time and well-being

There is currently much debate (and hype) surrounding smartphones and their effects on well-being, especially with regard to children and teenagers. We'll be looking at data from this recent study of English adolescents: Przybylski, A. & Weinstein, N. (2017). A Large-Scale Test of the Goldilocks Hypothesis. *Psychological Science, 28*, 204–215.

This was a large-scale study that found support for the "Goldilocks" hypothesis among adolescents: that there is a "just right" amount of screen-time, such that any amount more or less than this amount is associated with lower well-being. This was a huge survey study with data containing responses from over 120,000 participants! Fortunately, the authors made the data from this study openly available, which allows us to dig deeper into their results. And the question we want to expand on in this lab is whether the relationship between screen-time and well-being depends on the partcipant's (self-reported) sex. In other words, our research question is: **Does screen-time have a bigger impact on boys or girls, or is it the same for both?**

The dependent measure used in the study was the **Warwick-Edinburgh Mental Well-Being Scale (WEMWBS)**. This is a 14-item scale with 5 response categories, summed together to form a single score ranging from 14-70.

On **Przybylski & Weinstein's page for this study on the Open Science Framework**, you can find the participant survey, which asks a large number of additional questions (see page 14 for the WEMWBS questions and pages 4-5 for the questions about screen-time). Within the same page you can also find the **raw data**, which some of you might want to consider using for your research report.

However, for the purpose of this lab, you will be using local pre-processed copies of the data (`participant_info.csv`, `screen_time.csv` and 'wellbeing.csv, which you downloaded as part of the 'Pre-lab activities'.

Przybylski and Weinstein looked at multiple measures of screen-time, but again for the interests of this lab we will be focusing on smartphone use, but do feel free to expand your skills after by looking at different definitions of screen-time. Overall, Przybylski and Weinstein suggested that decrements in well-being started to appear when respondents reported more than one hour of daily smartphone use. So, bringing it back to our additional variable of sex, **our research question is now: Does the negative association between hours of smartphone use and well-being (beyond the one-hour point) differ for boys and girls?**

Let's think about this in terms of the variables. We have:

- a continuous outcome variable: well-being;
- a continuous* predictor variable: screen-time;
- a categorical predictor variable: sex.

Please note that well-being and screen-time are technically only **quasi-continuous** inasmuch as that only discrete values are possible. However, there are a sufficient number of discrete categories in our data that we can treat the data as effectively continuous.

Now, in terms of analysis, what we are effectively trying to do is to estimate **two slopes** relating screen-time to well-being, one for adolescent girls and one for adolescent boys, and then statistically compare these slopes. Sort of like running a correlation for boys, a correlation for girls, and comparing the two. Or alternatively, where you would run a regression (to estimate the slopes) but also one where you would need a t-test (to compare two groups). But the expressive power of regression allows us to do this all within a single model. Again, as we have seen building up to this lab, an independent groups t-test is just a special case of ordinary regression with a single categorical predictor; ANOVA is just a special case of regression where all predictors are categorical. But remember, although we can express any ANOVA design using regression, the converse is not true: we cannot express every regression design in ANOVA. As such people like regression, and the general linear model, as it allows us to have any combination of continuous and categorical predictors in the model. The only inconvenience with running ANOVA models as regression models is that you have to take care in how you numerically code the categorical predictors. We will use an approach called **deviation coding** which we will look at today later in this lab.

To complete this lab activity, you can use the R-script (`402_wk12_labAct1_template.R`) that you downloaded as part of the 'Pre-lab activities' as a template. Work through the activity below, adding relevant bits of code to your script as you go along.

3.4.1.2 Step 1: Background and set up

Before we get stuck in we need to set up a few things.

> **TASK**: Add code to clear the environment. **HINT**: `rm(list=ls())`

Next we need to tell R which libraries to use. We need `broom`, `car` and `tidyverse`.

> **TASK**: Add code to load relevant libraries. **HINT**: `library()`

Finally, read in the three data files; call the participant info `pinfo`; call the screen_time data `screen` and the well-being data `wellbeing`.

> **TASK*: Add code to read in the three data files. **HINT**: Use the `read_csv()` function.

3.4.1.3 Step 2: Checking the formatting

Given our research question and the information you have about the scores, provided above under 'Background' and from the OSF-webpage, is the data ready for use?

> **TASK**: Add code to look at the first few lines of each data frame.
> **HINT**: Use the `head()` function (or `tail()` function).

QUESTION 1: In which table is the variable corresponding to sex located and what is this variable called?

The 'source and analysis code.sps' file in the 'Data and Code' section on the **OSF-webpage** tells us how they coded the sex variable: 0 = female indicator and 1 = male indicator. It is worth exploring the OSF-webpage, to get used to foraging other files for these kinds of information, as they are not always clearly explained in a codebook or README. file.

For ease, lets recode the sex variable to reflect word labels of 'female' and 'male'. This doesn't change the order: R will still see female as 0, and male as 1 because female occurs before male in the alphabet. Add the code below to your script to do this. Don't forget to run it as well.

```
pinfo$sex <- ifelse(pinfo$sex == 1, "male", "female")
head(pinfo)
```

QUESTION 2: In what format is the well-being data (long or wide)? On how many participants does it include observations? And on how many items for each participant?

QUESTION 3: What is the name of the variable that identifies individual participants in this dataset? It is important to work this out as this variable will allow us to link information across the three data files.

3.4.1.4 Step 3: Data preparation - Aggregating the total well-being scores

We need to sum the 14 items of the well-being scale for each participant to create one well-being score per participant.

> **TASK**: To create one well-being score per participant add code to the script to do the following: first, transform the `well-being` variable from wide to long (using `pivot_longer()`); then, use `group_by()` to get scores for each participant and finally use `summarise()` to calculate a total well-being score, calling the new variable `tot_wellbeing`. Save all of this to an object called 'wb_tot'.

It is useful to calculate some descriptive statistics for the new variable `tot_wellbeing`.

> **TASK**: Calculate some descriptive statistics for `tot_wellbeing`. **HINT**: Use `summarise()` to calculate the mean, standard deviation, minimum and maximum values.

Finally, let's get an idea of the distribution of the new variable `tot_wellbeing`.

> **TASK**: Visualise the distribution in a histogram. **HINT**: Use `ggplot()` and `geom_historgram()`.

QUESTION 4: Is the distribution of well-being scores symmetrical, negatively skewed or positively skewed?

3.4.1.5 Step 4: Data preparation - Transforming screen time data

Great, so we have the well-being scores sorted out, we now need to think about the screen-time usage data and whether it is being used on a weekday or a weekend. As always, to get an idea of the data, it is often very useful to visualise the variables before proceeding with the analysis.

Before we can do this, we'll need to tidy these data up. Have another look at the `screen` data by using the `head()` function. You'll see that we have `Serial` in the first column (this is good), but in the following eight columns, we have columns for each type of activity (Comph, Comp, Smart, Watch) and the part of the week it took place (we and wk) combined. Instead, to be able to work with the data, we need two columns: one for the type of activity (we'll call it `variable`) and one for the part of the week (we'll call it `day`).

A second issue is that we need to alter the abbreviations Comph, Comp, Smart and Watch to reflect more descriptive text for each in plots.

Below are two chunks of code that represent these steps. In the next tasks you'll practise with taking a set of piped commands apart. The purpose of this is to get you used to "parsing" the code at the right places so that when you see piped commands in other people's code, you know how to break it down and find the relevant parts that you can use.

> **TASK**: In the code chunk below we use the `separate()` function to split the character strings already in the dataset. You know that with piped commands, there are chunks of code. Run the code first in its entirety and then pull each line apart to see how each function works on the data. Write a descriptive sentence for each function's role in the command. Don't forget to copy the chunk to your script and run it.

```
screen_long <- screen %>%
  pivot_longer(-Serial, names_to = "var", values_to = "hours") %>%
  separate(var, c("variable", "day"), "_")
```

> **TASK**: In the next code chunk we use the `dplyr::recode()` function with `mutate()` to relabel the separated names into understandable names that will be clear in plots. Again, run the code first in its entirety and then pull each line apart to see how each function works on the data. Write a descriptive sentence for each function's role in the command. Don't forget to copy the chunk to your script and run it.

```
screen2 <- screen_long %>%
  mutate(variable = dplyr::recode(variable,
                                  "Watch" = "Watching TV",
                                  "Comp" = "Playing Video Games",
                                  "Comph" = "Using Computers",
                                  "Smart" = "Using Smartphone"),
         day = dplyr::recode(day,
                             "wk" = "Weekday",
                             "we" = "Weekend"))
```

The code above has a new feature: the `dplyr::recode` part. This syntax – using the double colon – happens when there are many versions of a function with the same name. You can imagine that a function called 'recode' is immensely useful at the data wrangling stage of analysis. By using the name of the package, a double set of colons, followed by a function name, you are ensuring that R uses a particular version of the function, at that point only. This avoids having two or more packages loaded in your environment that sometimes do not play nicely together!

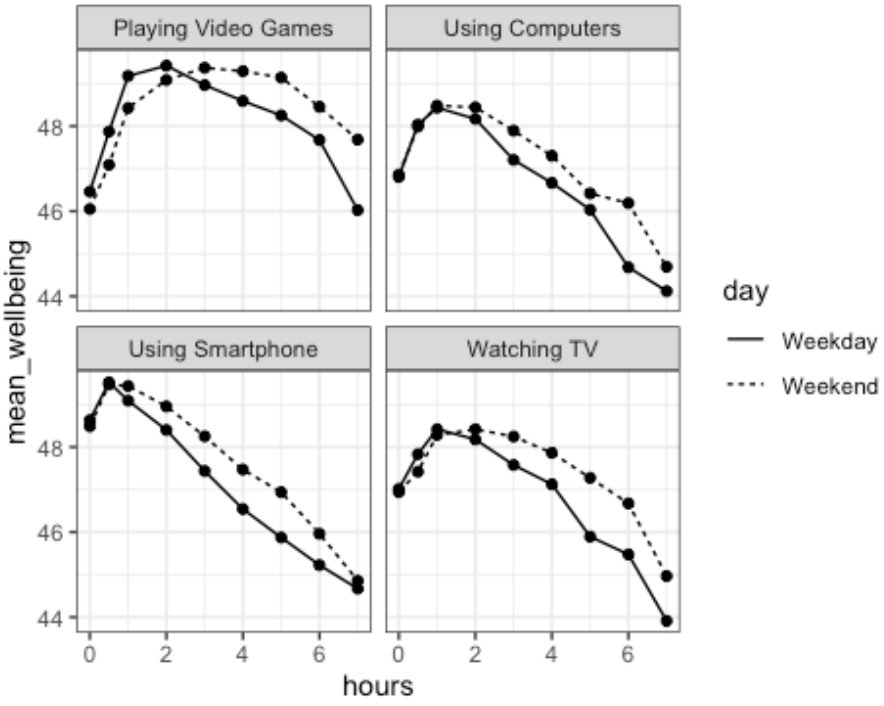
To be able to monitor that your code is performing as you want it to, you need to have in your mind an idea of how the data should look at the end of a code chunk. So stop a moment and be clear, discuss with your lab-mates if you feel like it and answer the following question.

QUESTION 5: What are the variables and the levels or conditions within each variable of screen2?

> **TASK**:Now join `wb_tot` and `screen_2` by participant and then group by the variables 'variable', 'day' and 'hours' and then calculate a 'mean_wellbeing' variable for each of the grouped levels. Save it in an object called 'dat_means'.**HINT**: Write separate lines of code for each action and then, when you know they all work, reformat them as a piped command; look back earlier in this code and last week for the functions that perform these actions.
>
> **TASK**: Now check that you have an object that is 72 observations

of 4 variables. You should have a mean wellbeing score for every level of the hours, over weekdays and weekends for each level of the four types of screen time (4 x 2 x 9)

Next, it is a good idea to visualise the mean well-being data as function of hours of screen-time for the different days (weekday vs. weekend) and types of screen (playing video games, using computers, using smartphone and watching tv). This is quite a complex graph. We'll go through creating it step-by-step, but let's first look at the end result:



Ok, that's what we are working towards.

TASK: Below, a chunk of code is presented. It is your task to fill in the x and y variables. **HINT**: Go back to the research question - which variable is for the x axis and for the y axis?

```
ggplot(dat_means, aes(x = , y = )) +
  geom_line() +
  geom_point() +
  theme_bw()
```

QUESTION 6a: What research question does this plot describe? Is it appropriate for the levels within the data?

TASK: Now, let's add a different linetype for each day (weekday

vs. weekend). Fill in the blanks in the code below.

```
ggplot(dat_means, aes(x = , y = , linetype = )) +
  geom_line() +
  geom_point() +
  theme_bw()
```

QUESTION 6b: What research question does this plot describe? Is it appropriate for the levels within the data?

Still not quite there.

> **TASK**: Fill in the blanks (for x, y and linetype) as before. Now have a good look at the code below. What has changed? Copy the code to your script and run it. Then, for each line write a sentence as a comment to describe its effect on the plot.

```
ggplot(dat_means, aes(x = , y = , linetype = )) +
  geom_line() +
  geom_point() +
  facet_wrap(~variable, nrow = 2) +
  theme_bw()
```

We add the `facet_wrap()` function here. You can check the `?facet_wrap()` help page for more information.

QUESTION 6: c. What does the `facet_wrap()` function do? Is this plot appropriate for the levels in the data?

3.4.1.6 Step 5: Calculating mean hours per day for smartphone use, for each participant

As mentioned at the beginning, in today's lab we'll focus on smartphone use. So looking at the bottom left of the figure we could suggest that smartphone use of more than 1 hour per day is associated with increasingly negative well-being the longer screen time people have. This looks to be a similar effect for Weekdays and Weekends, though perhaps overall well-being in Weekdays is marginally lower than in Weekends (the line for Weekday is lower on the y-axis than Weekends). This makes some sense as people tend to be happier on Weekends!

> **TASK**: Below is a set of comments that describe what the chunk of code that you need to write next does: use 'screen2' and then filter out the observations for 'Using Smartphone', and then group together each participant, and then summarise the mean hours calling it 'hours_per_day', save it in an object called 'smarttot'

> **TASK**: Now let's do it the other way around. Run the code below. Have a look at the structure of 'smart_wb' **HINT**: You can use the

`str()` function. What does the code do?

```
smart_wb <- smarttot %>%
  filter(hours_per_day > 1) %>%
  inner_join(wb_tot, "Serial") %>%
  inner_join(pinfo, "Serial") %>%
  mutate(sex = as.factor(sex))
```

QUESTION 7: What does the code do? Write a short paragraph, using the phrase "and then" to represent the pipes.

3.4.1.7 Step 6: More visualisation

We are now using only one small part of the data - smartphone use and its relationship with well-being over different durations of time. Before formally testing our research question, we can visualise the data and enquire about sex differences on the same plot - run each chunk of code below:

> **TASK** To further group the data, copy the code below to your script and run it. Look at the 'smart_wb_gen' dataframe. What has the code above done? Write a couple of sentences of description

```
smart_wb_gen <- smart_wb %>%
  group_by(hours_per_day, sex) %>%
  summarise(mean_wellbeing = mean(tot_wellbeing))
```

> **TASK**: Let's visualise these data.

```
ggplot(smart_wb_gen, aes(hours_per_day, mean_wellbeing, color = sex)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_discrete(name = "Sex", labels = c("Girls", "Boys"))+
  scale_x_continuous(name = "Total hours smartphone use") +
  scale_y_continuous(name = "Mean well-being score") +
  theme_bw()
```

QUESTION 8: Write an interpretation of the above plot in plain English.

3.4.1.8 Step 7: The regression model

In the steps 2 to 6 we've covered some pretty heavy-lifting data-wrangling. As it is so often the case that something like this is needed when working with real data, ti is really important to practise this. However, to ensure you also spend time on fitting the regression model and interpreting the output, you can choose to use the data-file smart_wb.csv to get started with that. It contains the data in a format that is the result of all the data-wrangling we did in steps 2 to 6. So,

download the smart_sb.csv data-file, put it in the folder that is your working directory and you're all set for running the regression model.

> **TASK**: Let's run the regression. Write code in your script in which you call your output 'mod', and use the data 'smart_wb' using the following formula, `lm(y ~ x1 + x2 + x1:x2, data)` to construct your regression model. Go back to the research question for your outcome and two predictor variables.
>
> **TASK**: Call and save the summary of your model as 'mod_summary'; then have a look at it.

Let's first look at the model as a whole:

QUESTION 9: What is the p-value for the overall model? Is it significant? What does this mean?

QUESTION 10: To two decimal places, what percentage of the variance in well-being scores does the overall model explain?

Now, lets look at the coefficients of our predictors:

QUESTION 11: Are the main effects of smartphone use and sex significant?

QUESTION 12: Which variable indicates the interaction between smartphone use and sex?

QUESTION 13: And is the interaction significant?

QUESTION 14: What is the most reasonable interpretation of these results?

The above model uses **treatment coding** (sometimes called dummy coding), for the sex variable. In a categorical variable with only two levels this means that one level is coded as 0 and the other level is coded as 1. In categorical variables with more than two levels, it works slightly differently.

> **TASK**: We can check that the sex variable is treatment or dummy coded with the following code:

```
contrasts(smart_wb$sex)
```

Because we have not explicitly told R about the labels for the sex variable, it has used level 0 as the reference level, hidden within the intercept term and level sex1 describes the difference (or slope) between level 0 and 1 or in this dataset from female to male.

QUESTION 15: Is being Male better for a person's well-being in the context of smartphone use than being Female?

Now let's look at **deviation coding**. There are other ways to code your categorical variables. One of them is deviation coding (also sometimes called sum coding). This effectively divides the difference of the values between your categorical levels by the number of levels so that each level can be compared to

one intercept that is central to them all rather than comparing levels to one reference level. It is like centering for a categorical level.

> **TASK** Use the code chunk below to: 1) add a variable to the smart_wb data that is a deviation coding of sex; 2) set the deviation coding (we'll labe it 'Sum' here for easy variable naming); and 3) look at the output for the sum-coded sex variable.

```
smart_wb <- mutate(smart_wb, sexSum = sex) # add a variable to the smart_wb data that is a deviat
contrasts(smart_wb$sexSum) <- contr.sum(2) # wet the deviation coding
contrasts(smart_wb$sexSum) # look at the output for the sum coded sex variable
```

Next, we'll run the regression again, using the sum-coded sex variable and we'll compare the outputs.

```
# Run the regression model again, using the sumcoded sex model and compare outputs
mod_S <- lm(tot_wellbeing ~ hours_per_day + sexSum + hours_per_day:sexSum, smart_wb)
mod_S_summary <- summary(mod_S)

# Compare the two model summary outputs
mod_summary
mod_S_summary
```

'sexSum1' is now the coefficient for sex and represents the change from the intercept value which now lies between the values for being Female and Male. Note how this coefficient is a negative.

The earlier model had a positive coefficient because the intercept described the reference group of the Girls, who on average begin at a lower well-being level than Boys (refer back to the scatterplot to verify this). Because the sum-coding has moved the intercept to a point that is the center of the difference between Boys and Girls, sexSum1 now describes the distance between the centre and a level of Sex.

Values for well-being in Girls are thus: Intercept + sexSum*+1 49.74 + (-1.61)*(+1)

Values for well-being in Boys are thus: Intercept + sexSum*-1 49.74 + (-1.61)*(-1)

with the Boys being higher in well-being…(remember a negative number multiplied by a negative number produces a positive number and a negative number multiplied by a positive number produces a negative number).

The interpretation of both model effects is the same, and if you look at the summary statistics, they are identical. Deviation coding effectively centers your categorical variables and helps with interpretation of interaction terms.

3.4.1.9 Step 8: Checking assumptions

Now that we've fit a model, let's check whether it meets the assumptions of linearity, normality and homoscedasticity. With regression models, you do this after you've actually fit the model.

Linearity Unlike when we did simple regression we can't use `crPlots()` to test for linearity when there is an interaction, but we know from looking at the grouped scatterplot that this assumption has been met.

Normality Normally we would test for normality with a QQ-plot and a Shapiro-Wilk test. However, because this dataset is so large, the Shapiro-Wilk is not appropriate (if you try to run the test it will produce a warning telling you that the sample size must be between 3 and 5000). This is because with extremely large sample sizes the Shapiro-Wilk test will find that any deviation from normality is significant. Therefore we should judge normality based upon the QQ-plot.

> **TASK**: Create a QQ-plot to check the residuals are normally distributed **HINT**: You can use the `qqPlot()` function. The residuals are stored in the 'mod' object you created earlier.

QUESTION 16: What do you conclude from the QQ-plot?

Homoscedasticity Here we have the same problem as with testing for normality: with such a large sample the `ncvTest()` will produce a significant result for any deviation from homoscedasticity. So we need to rely on plots again. To check for homoscedasticity we can use `plot()` from Base R that will produce a bunch of helpful plots (more information **here**:.

> **TASK**: Copy the code chunk below to your script and run it.

```
par(mfrow=c(2,2))              # 4 charts in 1 panel
plot(mod)                      # this may take a few seconds to run
```

The residuals vs leverage plot shows a flat red line so, whilst it isn't perfect, we can assume that with such a large sample size regression is still an appropriate analysis.

Multi-collinearity Finally, lets check for multicollinearity using the `vif()` function. Essentially, this function estimates how much the variance of a coefficient is "inflated" because of linear dependence with other predictors, i.e., that a predictor isn't actually adding any unique variance to the model, it's just really strongly related to other predictors. Thankfully, vif is not affected by large samples like the other tests. There are various rules of thumb, but most converge on a VIF of above 2 to 2.5 for any one predictor being problematic.

> **TASK**: Copy the code chunk below to your script and run it.

```
vif(mod)                       # Check for multi-collinearity
```

QUESTION 17: Do any of the predictors show evidence of multicollinearity? ## ANSWER: Yes, male and the interaction do. We'll talk about that more later in the module.

3.4.1.10 Step 9: Write up

QUESTION 18: How would you write up the results following APA guidance? You can choose whether you do so for the model using treatment coding or for the model using deviation coding.

3.5 Answers

When you have completed all of the lab content, you may want to check your answers with our completed version of the script for this week. **Remember**, looking at this script (studying/revising it) does not replace the process of working through the lab activities, trying them out for yourself, getting stuck, asking questions, finding solutions, adding your own comments, etc. **Actively engaging** with the material is the way to learn these analysis skills, not by looking at someone else's completed code…

The answers to the questions and the script containing the code will be available after lab session has taken place.