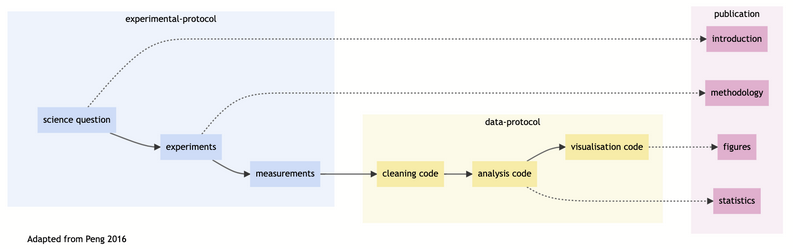
Computing Skills: Reproducible Figures Notes

* Scientific results and evidence are strengthened if results can be replicated and confirmed by other.
* Many scientific studies cannot be reproduced, with some part of this issue being in data analysis and code.
* Reproducible means authors have provided all the necessary data and computer code to rerun an analysis and recreate results.
* Replicable means a study arrives at the same findings as another study, collecting new data and new analyses.
* Reproducible science is helpful as usually the original researcher will have to redo work and carry out analysis at a later date.
  + Time is wasted redoing analyses because code cannot be read or understood after the fact, or regenerating figures as the code used to make them is lost.
* Also acts as a backup for work.
* Makes science more open, allowing for transparency, reusability, collaboration and accessibility (as well as accountability).
  + Do need to appropriately licence research outputs so prospective users understand reuse limitations.
* It is more common for code to be released as part of publishing and grant approval processes.
* 
* A diagram of a clean data analysis

  Description automatically generated
* Building data pipelines makes parts reusable for other purposes.
  + Can reuse code for things like making figures or running models.
* Functions can be really useful for making reusable code.
  + Also means if something is incorrect, only have to change something (and push to git) once, instead of having to change it multiple times.
* Renv allows for a record of packages and versions to be used in the code to be kept, making research more transparent.
* Every time install new package, need to update renv.
  + Install.packages In console OR
  + Renv::install([package\_name])

In this lesson, we've covered several key aspects of reproducible research and good coding practices in R:

1. Creating a structured project folder
2. Using relative file paths with the here package
3. Writing modular, reusable code with functions
4. Separating data cleaning steps into a dedicated script
5. Managing package dependencies with renv

* By implementing these practices, we've built a robust, reproducible data pipeline for our penguin data analysis project. This approach offers several benefits:
  + **Reproducibility**: Anyone can now run our analysis and get the same results.
  + **Collaboration**: Our well-organised project structure makes it easier for others to understand and contribute to our work.
  + **Maintainability**: Modular code and clear documentation make it easier to update or extend our analysis in the future.
  + **Portability**: Using renv ensures our code will work across different environments and over time.
* Remember, reproducible research is not just about following a set of rules, but about cultivating a mindset of transparency and rigour in your scientific work. Keep refining these practices and exploring new tools that can enhance the reproducibility of your work.
* By making your research reproducible, you're not only improving the quality and credibility of your own work, but also contributing to the broader scientific community by enabling others to build upon your findings more easily.

Week 4:

This lesson teaches us to:

1. Create publication-quality figures
2. Writing reuseable code for making figures
3. Save figures in a variety of formats
4. Avoid rewriting code for similar figures

NOTE: I am often having to tell my computer where to push files using

git push –set-upstream origin main

This tells my device how to push stuff onto git and where to put it.

Lol just found out you can only commit after pressing save first. Remember that babes xx

* Exploratory figures show raw data as is.
  + Allow us to check data looks as expected.
    - Use things like boxplots, scatter graphs (without line of best fit), histograms etc.
* Using here function makes code more private, more reproducible.
  + This is a relative path, specific to the users computer, meaning other people can use your code more efficiently.
* Using geom\_jitter by itself can be problematic as it prescribes a random x-value to the jitter.
  + Can tell computer to generate x-value in same way every time (random seed)
  + Important for people using Bayesian stats where they need to be able to reproduce stuff.
* Alpha increases transparency of dots.
* Use colour blind friendly colourschemes.
* c= concatenate. Tells r to put everything in together.
* Could make function and put in separate script, to tell R exactly how to make boxplot.
* Can overwrite inside functions as they have transient existence.
* No need to keep rewriting code for figures for diff purposes e.g a4 and poster, just edit code in r.
* To change font sizes relative to figure, can change scaling to make stuff scale up.
* If increase size, increase scaling.
* Save figures as .svg or .pdf. Image vectors.
  + Make images crisper- saved every line as line not number of pixels.
  + Saves every element of image as a shape, meaning can zoom in as much as possible.
* If need to make 3 versions of boxplot, can code and save as 3 diff files.

Git:

* Programming language made to keep track of files, and keeping files safe.
* GitHub is product acting as project repository.
* If trying to keep something safe, can use github to track version history and ensure stuff is kept safe.
* Important for transparency, especially given high standards scientists are held to.
* LF assignment is about collaboration and using code generated by other people.
* Commit, push and pull (pull from someone else, push sends to github, commit keeps on your machine).
* Create github repository to send files to.