

Reproducibility

Recap So Far

RStudio

- The Editor is for static code like scripts or R Markdown documents
- Send code to the Console to run it
- The Console can be used for quickly testing code on the fly
- R code goes within what is called a chunk (the gray box with a green play button)

□ [RStudio Cheatsheet](#)

Recap So Far

Basic R

- Use `c()` to **combine** vectors
- Use `<-` to save (assign) values to objects
- if you don't use `<-` to reassign objects that you want to modify, they will stay the same
- `length()`, `class()`, and `str()` tell you information about an object
- `head()` and `tail()` can also help you inspect an object

What's Reproducibility

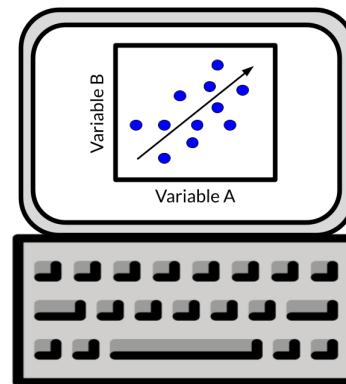
Reproducibility:

a different analyst re-performs the analysis with
the **same code** and
the **same data** and obtains
the **same result**.

Patil, Peng, Leek (2016) <https://www.biorxiv.org/content/10.1101/066803v1>

Content adapted from [Candace Savonen](#).

My data analysis is showing a pattern that is very informative for the ongoing research in my field.

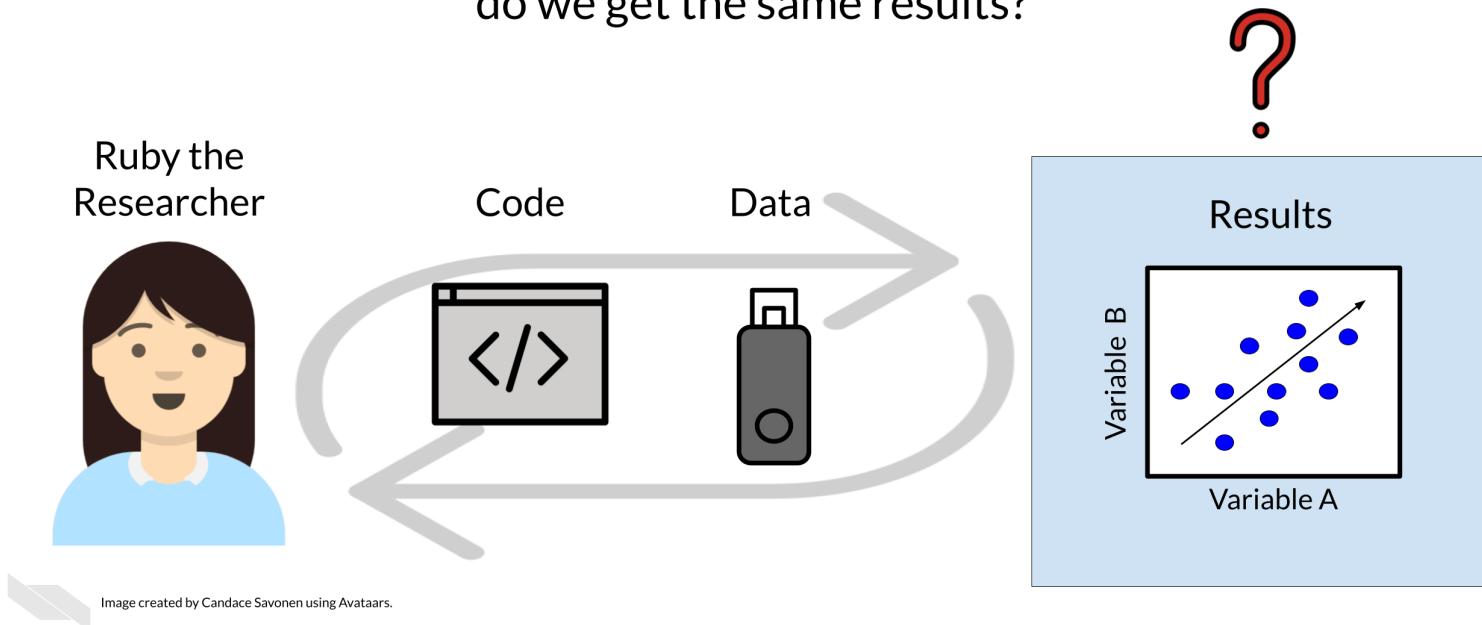


Ruby the Researcher

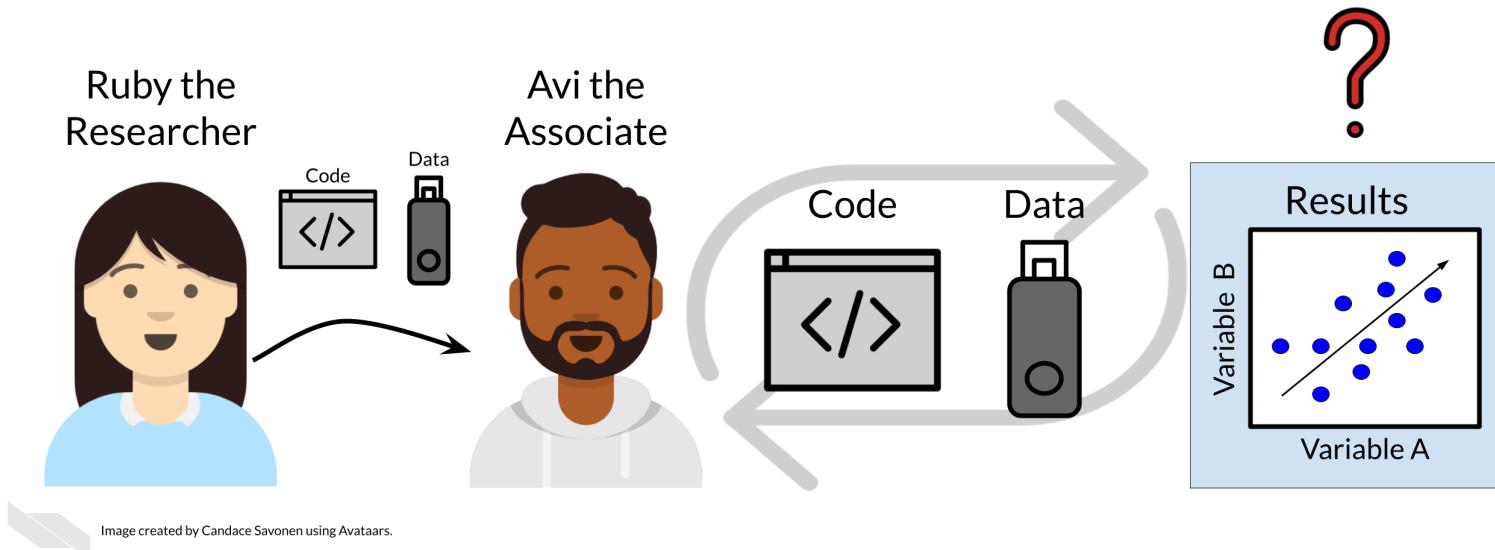


Image created by Candace Savonen using Avataars.

Repeatable: keeping everything the same but repeating the analysis - do we get the same results?

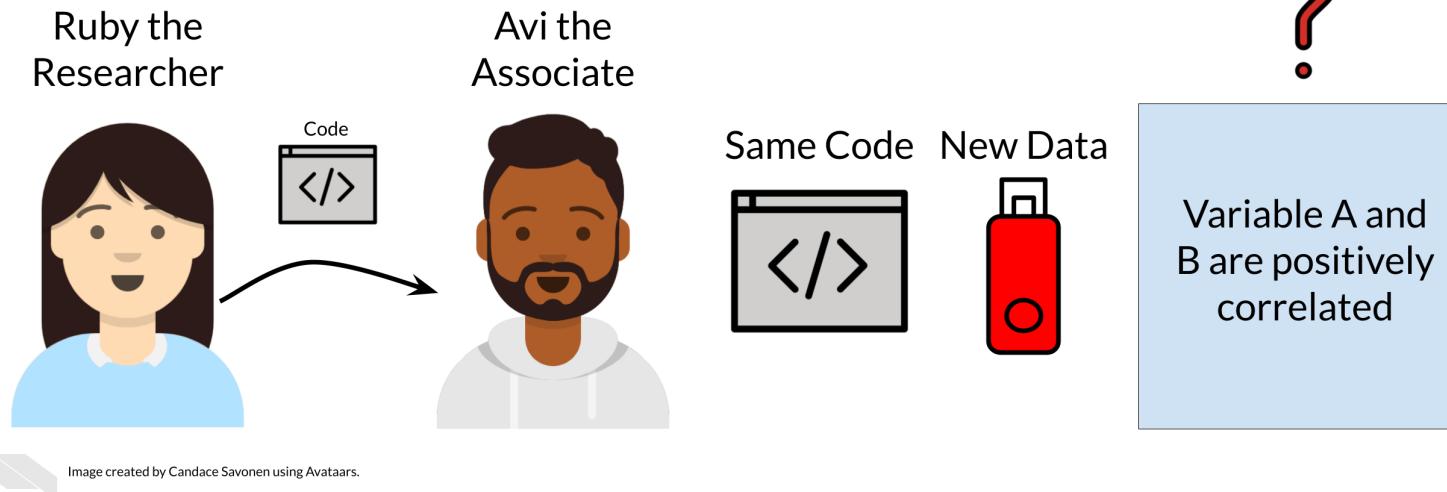


Reproducible: using the same data and analysis but in the hands of *another researcher* - do we get the same results?

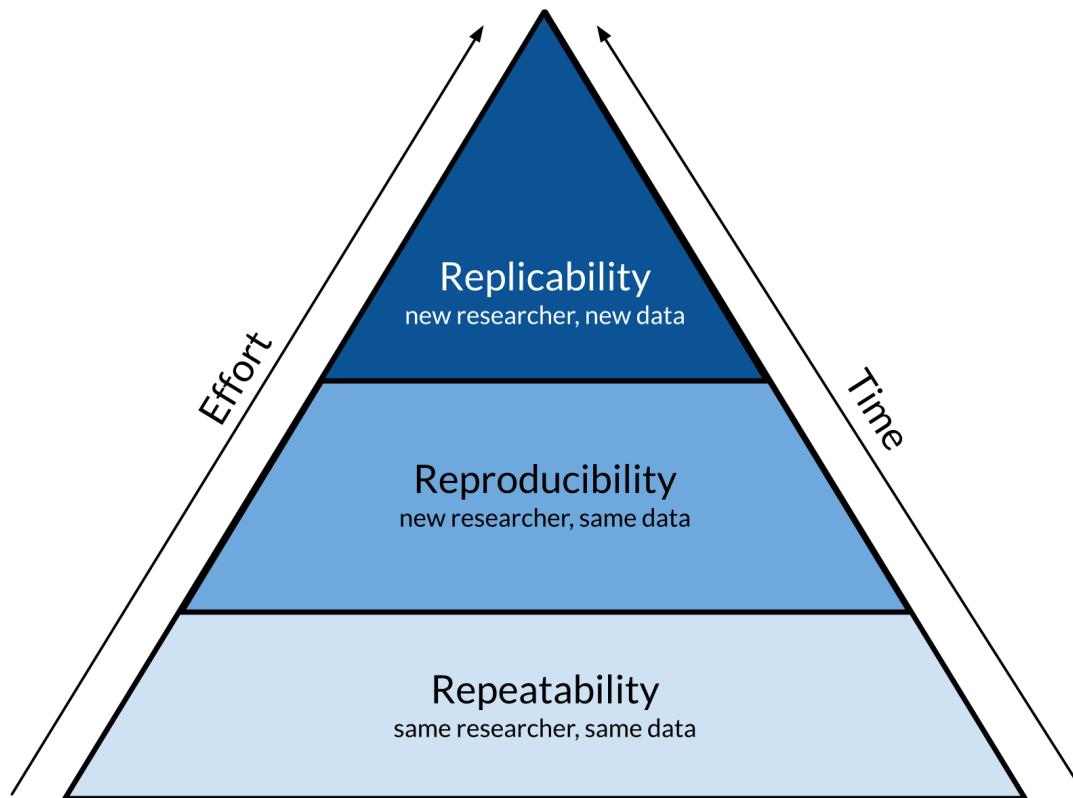


Replicable:

with new data do we obtain the same inferences?



Reproducibility vs Repeatability vs Replicability



Based off of a figure from Essawy et al, 2020 <https://doi.org/10.1016/j.envsoft.2020.104753>

Why Reproducibility is important...

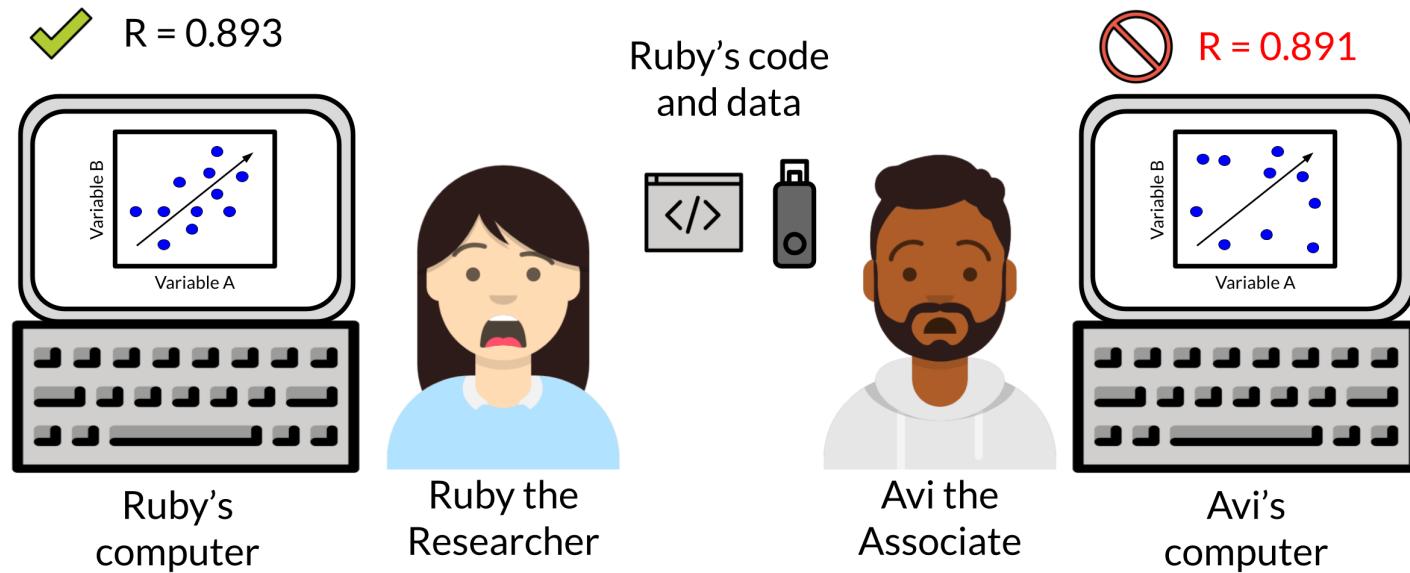
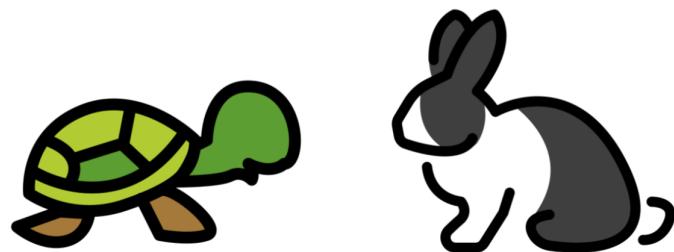


Image created by Candace Savonen using Avatars.

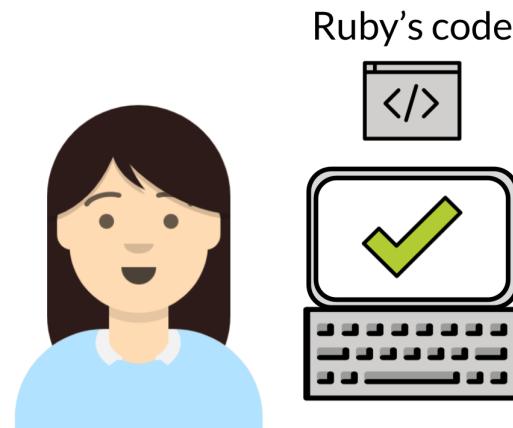
It's worth the wait

Reproducibility is a tortoise's game - it's an incremental and slow process *but it has high payoffs!*



Reproducibility can also be for your future self!

Now Ruby



Future Ruby

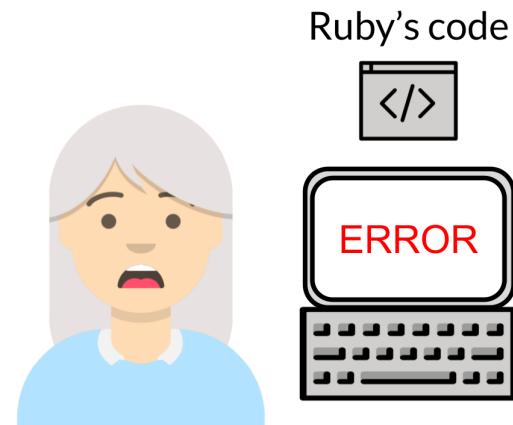


Image created by Candace Savonen using Avataars.

The process

Step 1) Get your code to work once

Step 2) Get your code to work reliably for you

Step 3) Get your code to work for someone else

R Markdown

R Markdown notebooks are a handy tool for reproducibility!



R Markdown lets you test your work

Working from this notebook allows me to interactively develop on my data analysis and write down my thoughts about the process all in one place!



RMarkdown is conducive to interactive development!

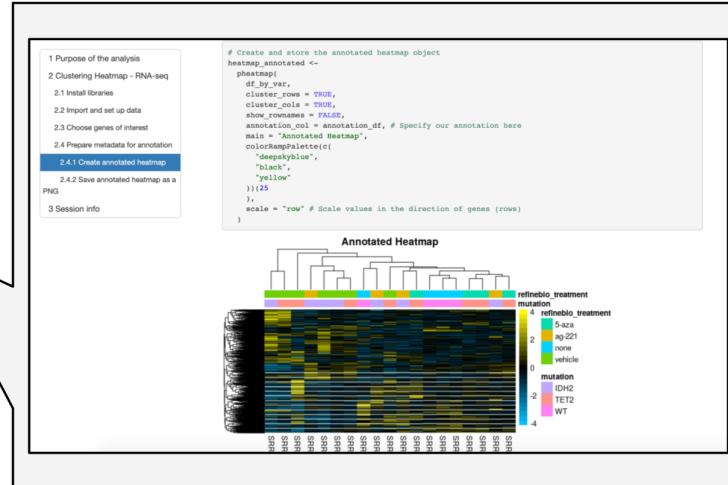
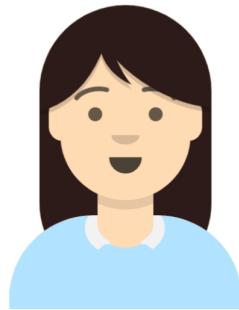


Image created by Candace Savonen using Avataars.

R Markdown allows you to more clearly show what you did

Avi, here's some output from this scientific notebook I've been developing from!

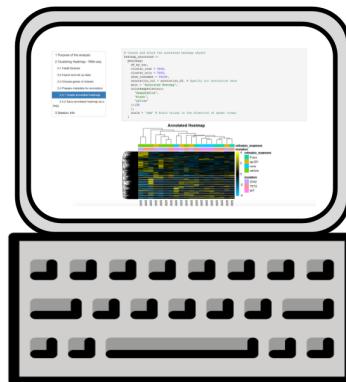


Ruby the Researcher

This is so easy to follow and read, even though I didn't write the code. Thanks for sharing your exciting results!



Avi the Associate



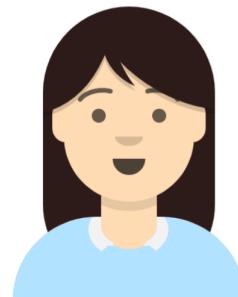
RMarkdown creates easily shareable output!



Image created by Candace Savonen using Avataars.

R Markdown makes it easier to update code and see results

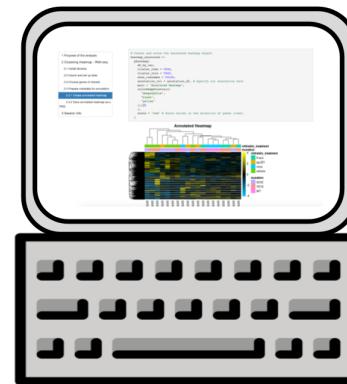
Yay! I just got the data for 5 more samples. Because of my handy notebook set up, I can easily call one command and re-run the analysis so it is updated with the new samples included!



Ruby the
Researcher



Image created by Candace Savonen using Avataars.

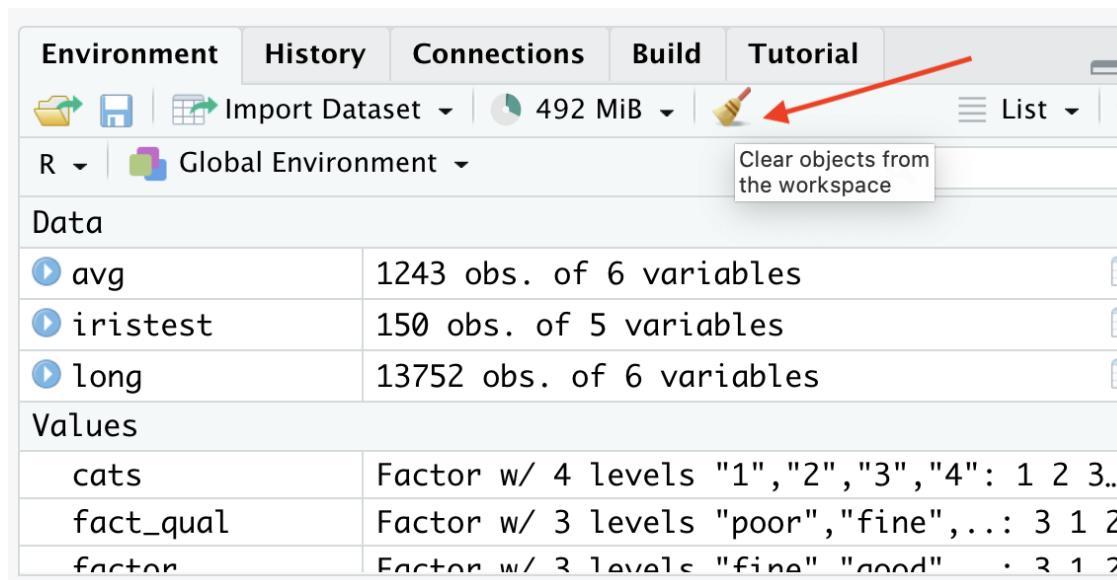


**RMarkdown is handy for
creating updateable reports!**

Clean your environment

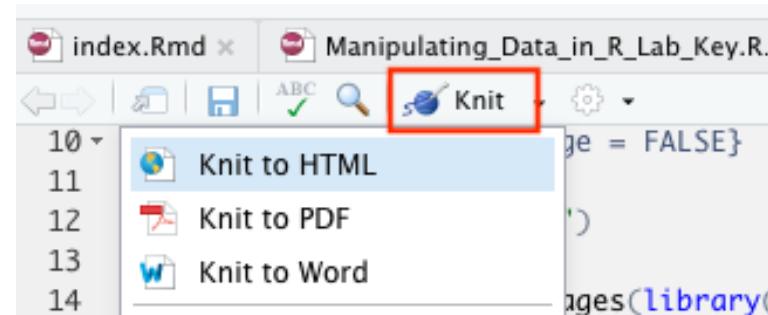
Regularly cleaning your environment and trying your code again, can help ensure that your code is running as expected.

Occasionally we might forget to save a step of our code in our R Markdown file that we ran only in the console. This will help us figure that out.



Check if your file knits regularly

Regularly checking if your file knits will help you spot a missing step or error earlier when you have less code to try to identify where your code might have gone wrong.



Tell your future self and others what you did!

Provide sufficient detail so that you can understand what you did.

Need random numbers to stay consistent?

Use `set.seed()` : sets the starting state for the random number generator (RNG) in R.

```
[1] 10 7 6 3 1 2 5 9 4 8
```

R Markdown syntax

Before:

```
# Header - biggest font created by hashtag and space
## SubHeader Second Biggest created by 2 hashtags and space

**bold** text
*italicized* text

`code` referenced outside of a chunk needs backticks
```

After knit:

Header - biggest font created by hashtag and space
SubHeader Second Biggest created by 2 hashtags and space

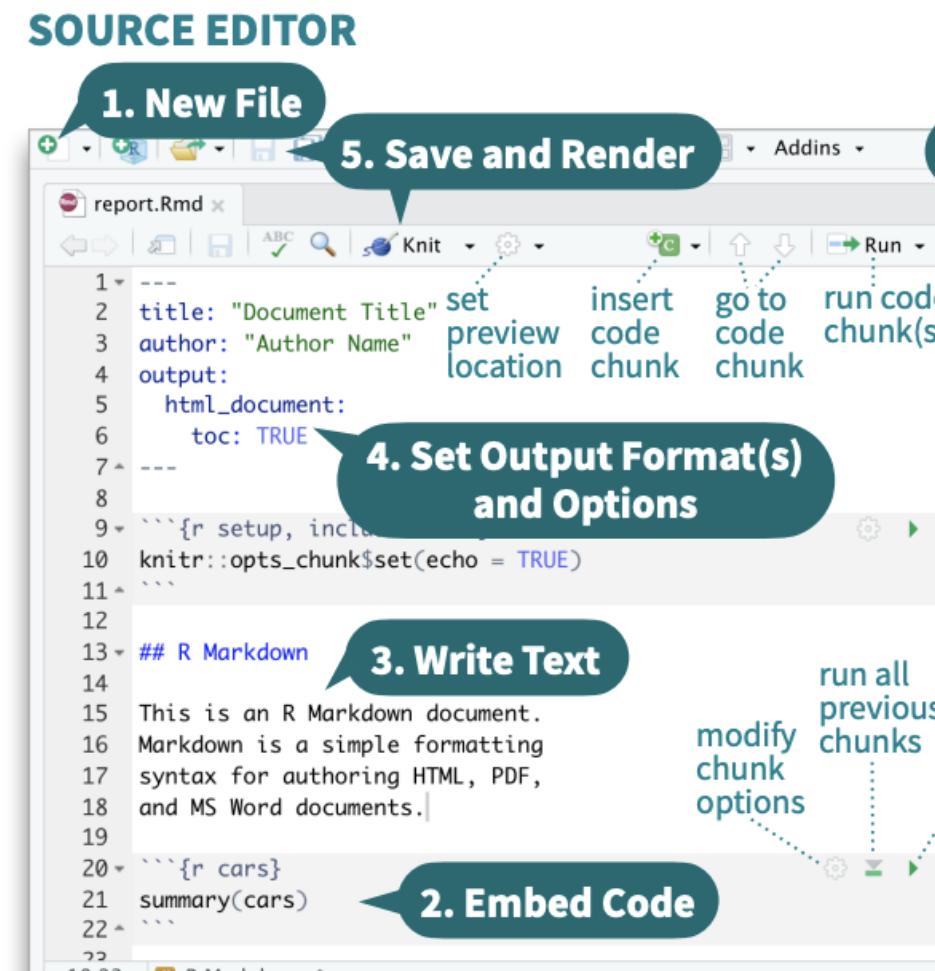
bold text *italicized* text
`code` referenced outside of a chunk needs backticks

R Markdown syntax

Go to Help > Cheat Sheets > R Markdown Cheat Sheet (which will download it)

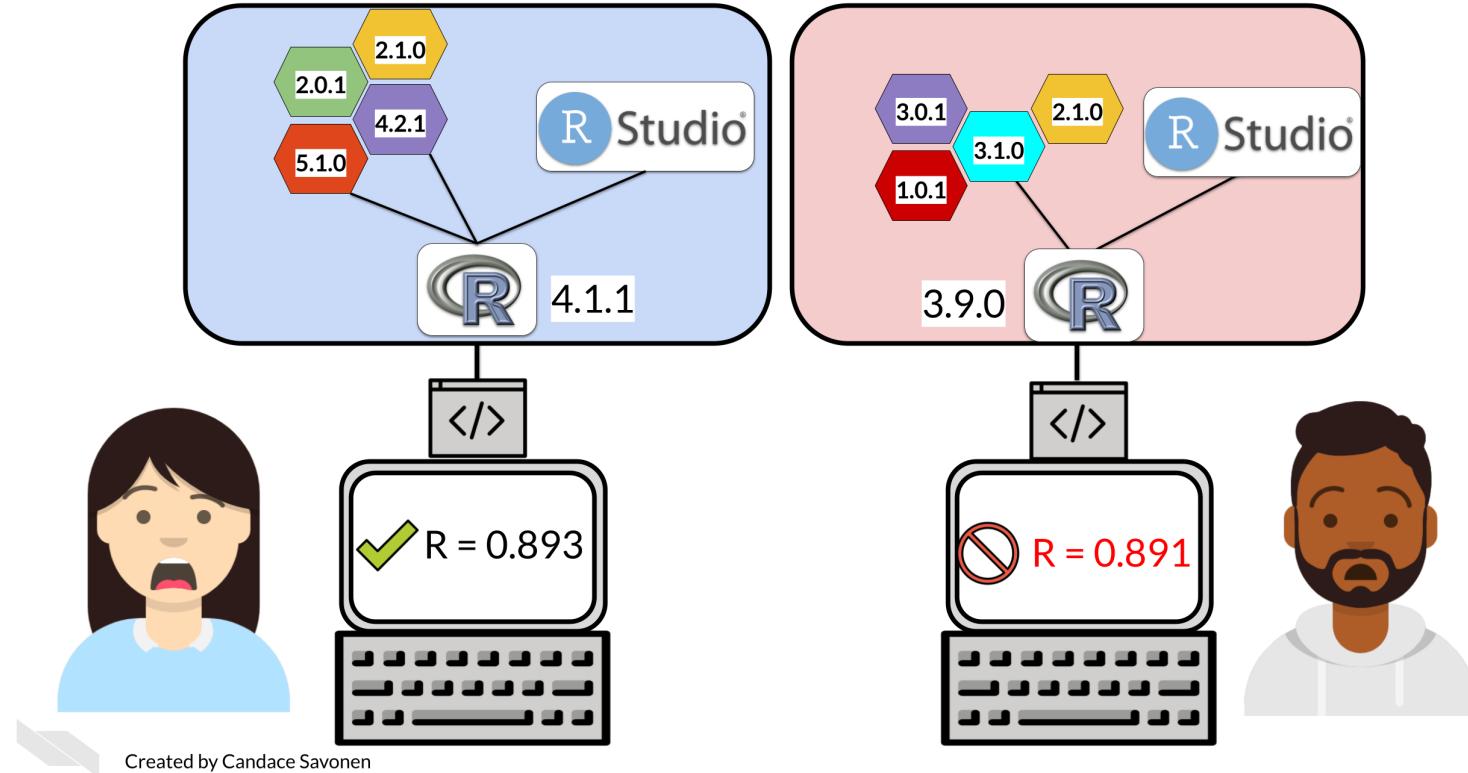
Or checkout Help > Cheat Sheets > R Markdown Reference Guide

Or checkout the [Class Website!](#)



Versions matter

Ruby's local computing environment Avi's local computing environment



Session info can help

Ruby's session info print out

```
R version 4.0.2 (2020-06-22)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04.2 LTS

Matrix products: default
BLAS/LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblas-r0.3.8.so

locale:
[1] LC_CTYPE=en_US.UTF-8
[4] LC_COLLATE=en_US.UTF-8
[7] LC_PAPER=en_US.UTF-8
[10] LC_TELEPHONE=C
[1] LC_NUMERIC=C
[2] LC_MONETARY=C
[3] LC_NAME=C
[4] LC_MEAS=C

attached base packages:
[1] stats      graphics   grDevices utils      datasets   methods    base

other attached packages:
[1] rmarkdown_2.4

loaded via a namespace (and not attached):
[1] rstudioapi_0.11  knitr_1.30    magrittr_1.5    hms_0.5.3    tidyselect_1.0.0
[6] R6_2.4.1        rlang_0.4.7   fansi_0.4.1    dplyr_1.0.2   tools_4.0.2
[11] xfun_0.18       sessioninfo_1.1.1 tinytex_0.26  cli_2.0.2    withr_2.3.0
[16] htmltools_0.5.0 ellipsis_0.3.1 assertthat_0.2.1  memoise_2.2.1 digest_0.6.25
[21] tibble_3.0.3    lifecycle_0.2.0 crayon_1.4.1   evaluate_0.14
[26] vctrs_0.3.4    glue_1.4.2    evaluate_0.14
[27] generics_0.0.2  iconlite_1.7.1  nkroconfig_2.0.3
```

R version 4.0.2 vs 4.0.5

Different operating systems!

Avi's session info print out

```
R version 4.0.5 (2021-03-31)
Platform: x86_64-apple-darwin17.0 (64-bit)
Running under: macOS Big Sur 10.16

Matrix products: default
LAPACK: /library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats      graphics   grDevices utils      datasets   methods    base

other attached packages:
[1] rmarkdown_2.10

loaded via a namespace (and not attached):
[1] lemmatize_0.1.2   BiocManager_1.30.16 compiler_4.0.5    magrittr_2.0.1
[5] fastmap_1.1.0    htmltools_0.5.2    tools_4.0.5     yaml_2.2.1
[9] tinytex_0.33     knitr_1.33      digest_0.6.27  xfun_0.25
[13] rlang_0.4.11     evaluate_0.14
```

rmarkdown 2.4 vs 2.10

If Avi and Ruby have discrepancies in their results, the session info print out gives a record which may have clues to why that might be!



Image by Candace Savonen

Lab 1

□ [Class Website](#)

□ [Lab](#)

More resources

These are just some quick tips, for more information:

- [Reproducibility in Cancer Informatics course](#)
- [Advanced Reproducibility in Cancer Informatics course](#)
- [The RMarkdown book](#)
- [Jenny Bryan's organizational strategies.](#)
- [Write efficient R code for science.](#)

Summary

To help make your work more reproducible:

- Use RMarkdown
- Clean your environment regularly
- Check the knit of your RMarkdown regularly
- Tell your future self and others what you did!
- Print session info!

□ [Class Website](#)



Image by [Gerd Altmann from Pixabay](#)