Workflows and best practices for collaborative coding in R

Dr Caitie Kuempel

@cdkuempel

Housekeeping and disclaimers

- Assuming you are familiar with R, Rstudio and Github
- More lecture and less tutorial today
- Please speak up and comment in the chat window with comments, questions, useful tips
- I'm a conservation scientist – I learned everything I know from you and twitter
  - Also this means I'm speaking from an academic perspective
- I'm still learning, but always trying to improve ☺
- Special thanks to Dr Alexa Fredston (@Afredston) for providing materials for this talk

Outline

- Tools, tips and tricks for:
  - Organizing your data
  - Coding workflows, best practices, reproducibility
  - Project management
  - Collaborating across skill levels and programming languages

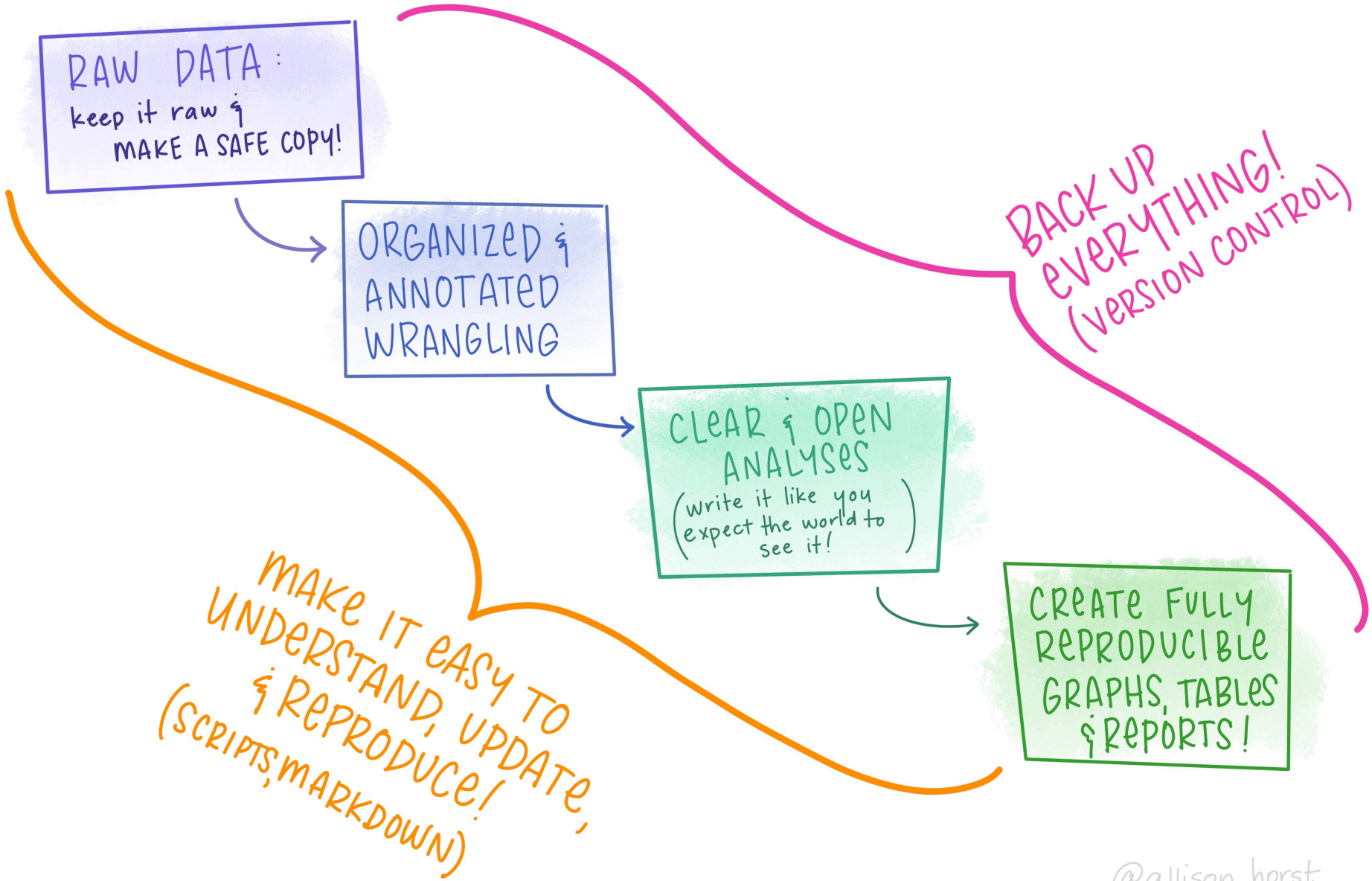
Speaking of Twitter….

- Twitter has an amazing R community
  - ”Tidy Tuesday”
  - Ask questions and get answers
  - Learn about new packages, useful tips, etc.
- @RLadiesBrisbane
- @hadleywickham (Chief Scientiest at Rstudio)
- @JennyBryan (Software engineer atRstudio)
- @Allison_horst
- For how to get started see here

Why does it matter?

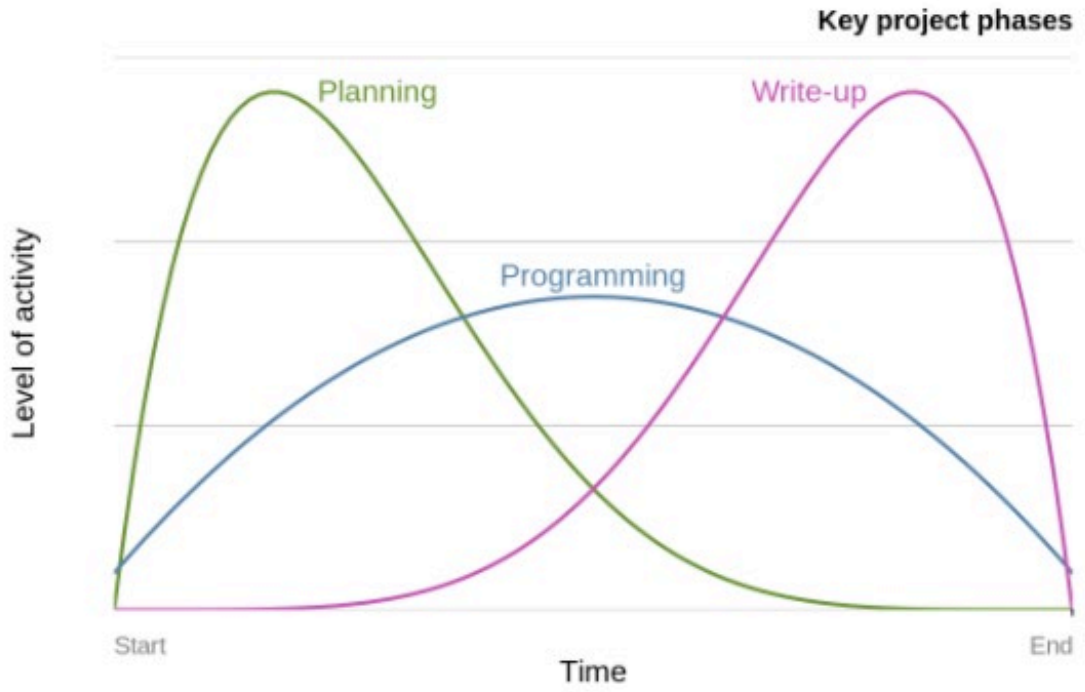
- A good project layout will ultimately make your life easier:
  - Makes it easier to understand the pipeline from source data to final product
  - Helps ensure the integrity of your data
  - Makes it simpler to share your code with someone else
  - Allows you to easily upload your code with your manuscript or project
  - Makes it easier to pick the project back up after a break

"Your greatest enemy is yourself four months ago" – every grad student ever


RAW DATA:
keep it raw &
MAKE A SAFE COPY!
ORGANIZED & ANNOTATED WRANGLING
CLEAR & OPEN ANALYSES
(write it like you expect the world to see it!)
CREATE FULLY REPRODUCIBLE GRAPHS, TABLES & REPORTS!
BACK UP EVERYTHING! (VERSION CONTROL)
MAKE IT EASY TO UNDERSTAND, UPDATE, & REPRODUCE! (SCRIPTS, MARKDOWN)
@allison_horst

Organizing your data

- Anticipate your code and data will be published (or shared widely) with every completed project/paper
  - Deposit in archives
    - FigShare
    - Knowledge Network for Biocomplexity (KNB)
- Put some thought into it – map out steps before you start
  - Write in code headings
- Treat raw data as read only
  - Raw_data directory that is never modified
- Web scraping/APIs



Organizing your data

- Web scraping/APIs
  - Allows others to reproduce your results directly from scripts without requiring any additional data files
  - Several R packages allow you to communicate with services to access data from R
    - Some conservation related examples
      - Rerddap
      - Rfishbase
      - Rnaturalearth
      - Rtweet
      - rredlist
      - Others?
  - You can also write your own API in R

Organizing your data

- Best practices
  - Use projects – one folder per project
    - Keeps everything organized, self-contained, increases reproducibility
    - Can work on multiple projects at once
  - Rmarkdown
    - Pros
      - Easily combines text, code and outputs
      - Allows results of R code to be directly inserted into formatted documents
      - Easy to use
      - You can even write entire papers or theses – check out bookdown
    - Cons: Can be clunky with large analyses
  - Use portable file paths – no absolute paths!
    - here() package

here() package

- Do you have setwd() in your scripts?
  - This makes your script fragile, hard-wired to exactly one time and place.
- here() sets the file.path() to the top-level of the current project

```
library(here)
#> here() starts at /Users/IRE/Projects/NICAR/2018/workflow
here()
#> [1] "/Users/IRE/Projects/NICAR/2018/workflow"
```

```
here("Test", "Folder", "text.txt")
#> [1] "/Users/IRE/Projects/NICAR/2018/workflow/Test/Folder/test.txt
cat(readLines(here("Test", "Folder", "text.txt")))
#> You found the text file nested in these subdirectories!
```

Organization your data

- As a general rule:
  - 1 script = ~1 task (or no more than one section of Methods)
  - List decision points/user choices at the top so it’s obvious what they are
  - Some examples from my work:
    - A bad script
    - A better script

Organizing your data

- Best practices
  - File organization
    - workflow
    - usethis
    - startR

At minimum

```
name_of_project
|--data
    |--2017report.csv
    |--2016report.pdf
    |--summary2016_2017.csv
|--docs
    |--01-analysis.Rmd
    |--01-analysis.html
|--scripts
    |--exploratory_analysis.R
|--name_of_project.Rproj
|--run_all.R
```

Optimal

```
name_of_project
|--raw_data
    |--WhateverData.xlsx
    |--2017report.csv
    |--2016report.pdf
|--output_data
    |--summary2016_2017.csv
|--rmd
    |--01-analysis.Rmd
|--docs
    |--01-analysis.html
    |--01-analysis.pdf
    |--02-deeper.html
    |--02-deeper.pdf
|--scripts
    |--exploratory_analysis.R
    |--pdf_scraper.R
|--name_of_project.Rproj
|--run_all.R
```

workflowR

- A R package that helps scientists organize their research in a way that promotes effective project management, reproducibility, collaboration and sharing of results

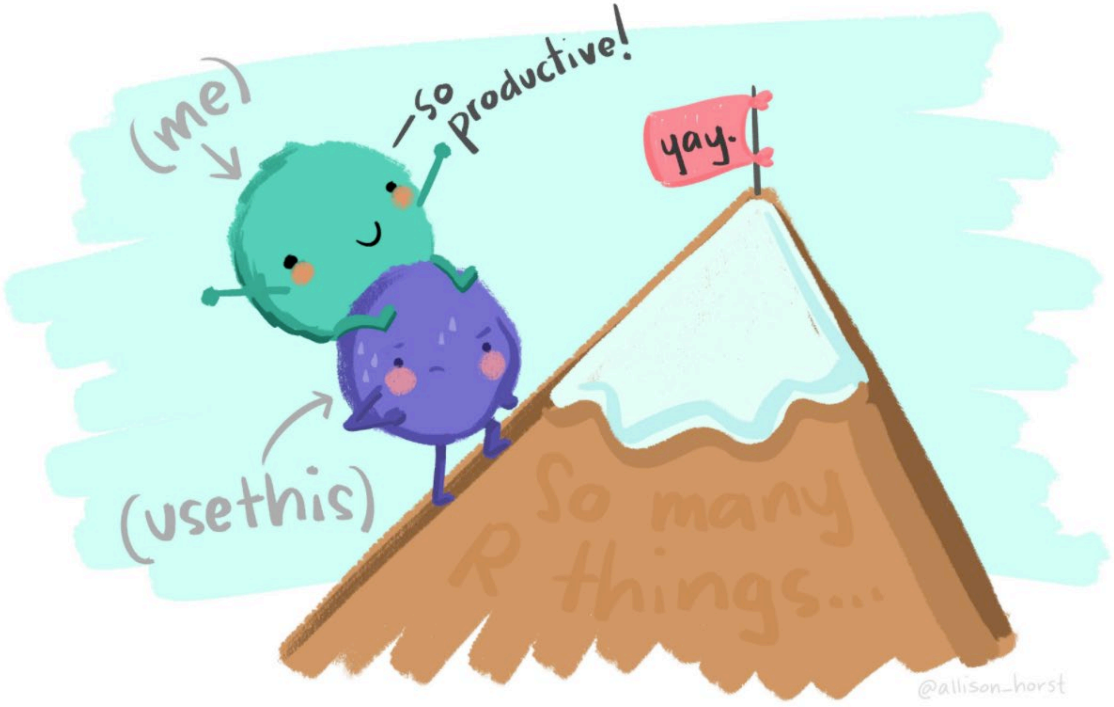
```
library("workflowr")
## This is workflowr version 1.6.2
## Run ?workflowr for help getting started
```

```
# Replace the example text with your information
wflow_git_config(user.name = "Your Name", user.email = "email@domain")
```

```
wflow_start("myproject")
```

usethis

- For package development
- Automates repetitive tasks that arise during project setup and development, both for R packages and non-package projects
  - create_project()
  - use_git()
  - use_github()



Project management and Collaboration

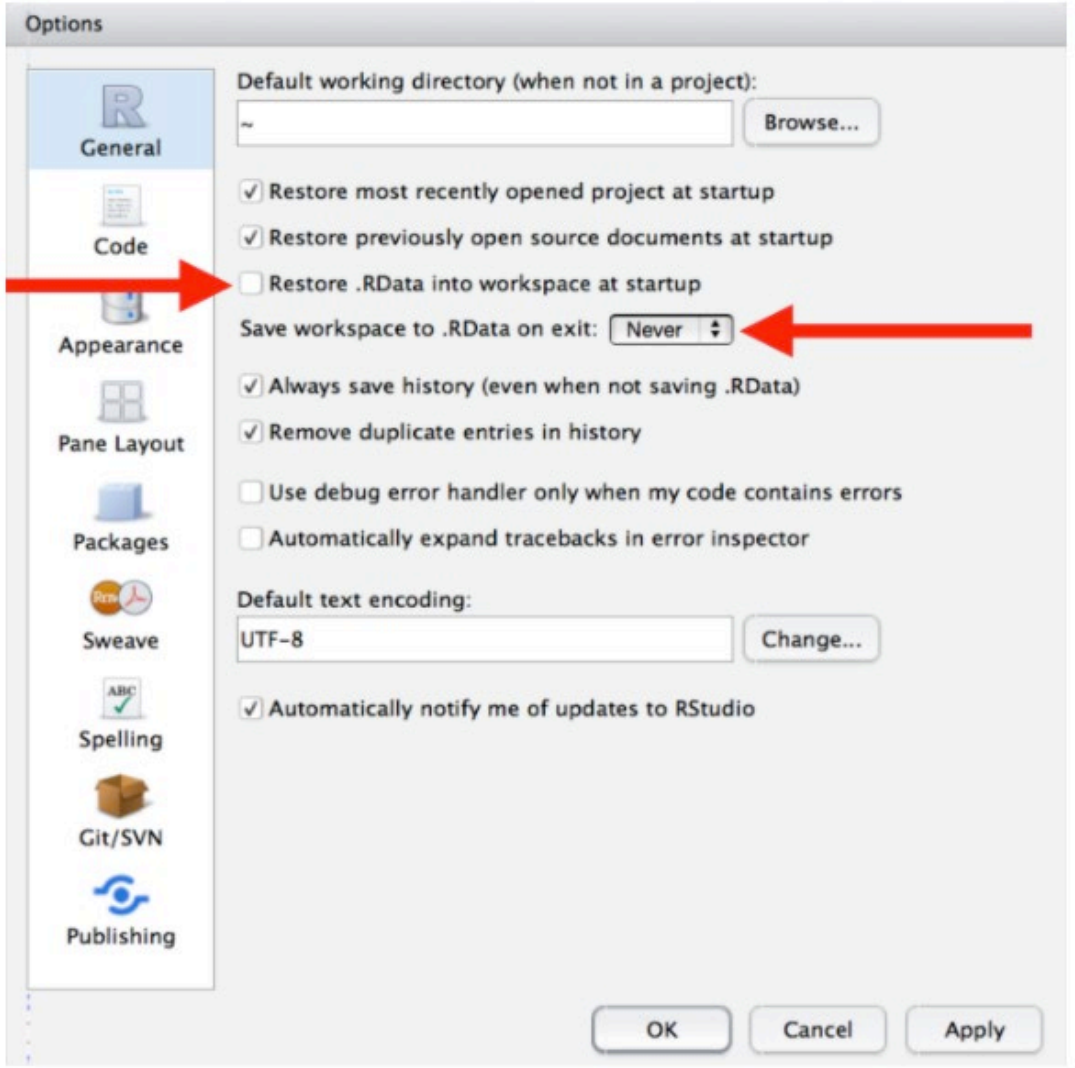
- Top tips for efficient collaboration
  - Describe what each code does
    - Readme.md
  - Have a consistent coding style
    - R style guides
      - http://adv-r.had.co.nz/Style.html
    - Be concise, clear and consistent
      - With comments and formatting!
    - Avoid repeating yourself!
      - Functions, “source” from other scripts - **example**
  - Think carefully about your comments and keep them up to date
  - Use version control whenever possible
  - Use informative commit messages
  - Don’t be afraid to ask for feedback from colleagues

Project management and Collaboration

- Describe what each code does
  - Readme.md
- Have a consistent coding style
  - R style guides
    - http://adv-r.had.co.nz/Style.html
  - Be concise, clear and consistent
    - With comments and formatting
- Avoid repeating yourself!
  - Data import (iterate over list of file names)
  - Data cleaning/tidying
  - Applying many models
  - Generating and saving plots
- Simple functions can be within a script, complex or functions used in multiple scripts can live in a functions folder/script.
- Apply functions using apply, purrr or for loops
  - Parallel processing = mclapply or pbmclapply

Project management and Collaboration

- Do not save workshops to .Rdata
  - Short-term pain – R will not remember the results of the code that you ran last time
  - Long-term gain – forces you to capture all important interactions in your code
- Use version control!
  - Github issues
    - Useful whether or not everyone codes
  - Github projects



Project management: day to day

- Try to pause coding when a task is complete
- Leave a to-do list for others (and yourself) of what still needs to happen for the script to be functional
- Pull + commit + pull + push frequently
- Restart R frequently (ensures right packages are loaded)

Project management and Collaboration

- Agree on shared practices for data storage, script development, communication, etc. at the start of a project and document them in a Google doc

- Example

Project management and Collaboration

- Check if they have a “how we work document” and/or set some basic guidelines and follow them
- Editing other people’s code
  - Be constructive, be nice!
  - Check if they have any contribution guidelines
  - Fix bugs if you see any
  - There are many, many, many ways to do the same thing in R – all of them “correct”, some more efficient (runtime/key strokes) or easier to understand

Project management and Collaboration

- Collaborating across different open-source programming languages
  - Extra important to have good communication and division of tasks
  - Can still use git for version control of scripts
  - Write out and read in data in flexible formats that translate across the languages you are using
  - R packages
    - Reticulate lets you run Python code from Rstudio (still need to understand Python a bit)
    - Other ideas?

Collaboration

- Collaborating across different open-source programming languages
  - Many scientists feel strongly about using ArcGIS, Matlab, etc.
  - Best practices about version control and data management still apply
  - Half a reproducible project is better than none
  - Make data to reproduce results publicly available when at all possible

Collaboration

- For collaborators who code, but aren't using best practices
  - Share some of the tools/tips you have learned here
  - Offer to help set up and maintain reproducible workflow for/with them
  - A little organization and planning to go a long ways – it doesn't have to be hard to use tools and it doesn't have to be "perfect" to contribute

- For collaborators who don't really code:
  - Use GitHub for communication (Issues) and sharing data and figures
  - Rmarkdown/Shiny to show code and results
  - Rmarkdown interface to Microsoft Word (has anyone tried this?)
  - Google Docs > Microsoft Word
  - People are worth more to projects than tools
  - Other thoughts?

Collaboration

- Getting people on board
  - Share resources and offer to help
  - Start coding clubs/hacky hours in your organization

Supercharge your research: a ten-week plan for open data science

Researchers share tips for transforming your group with open data science and teamwork.

Our path to better science in less time using open data science tools

Julia S. Stewart Lowndes ✉, Benjamin D. Best, Courtney Scarborough, Jamie C. Afflerbach, Melanie R. Frazier, Casey C. O'Hara, Ning Jiang & Benjamin S. Halpern

Nature Ecology & Evolution **1**, Article number: 0160 (2017) | Cite this article

Questions?

- Questions for me???
- Some questions for you!
  - What has been your biggest hurdle for workflows and collaborative coding in R? How are you handling it now? What might you try differently in the future?
  - Do you normally make your code/data available when a project is complete? If yes, what tools do you use? If no, what is the biggest barrier?
  - Any tools/tips/tricks that you love that weren't mentioned in the talk?
  - How many people are usually working on a project with (or planning to?) at one time? Any success stories to share with the group?
  - What area (e.g., data organization, project management, collaboration, etc.) do you want to try to improve the most from your workflow?

Thanks!

- Follow me on twitter @cdkuempel
- Get in touch if you would like to present at a future Rladies meeting or have a topic you would like to see covered