Introduction to Reproducible Research in Bioinformatics

Nan Xiao @road2stat

CRI Bioinformatics Workshop

About me

- http://nanx.me
- Statistics background
- Previous experience in statistical machine learning , systems pharmacology, translational bioinformatics
- R developer (7 R/Bioconductor packages; 4 web applications; 4 translated books)

Agenda

- Concept
- Principles
- Tools
- Challenges

We love copy & paste

Data updated today …

Concept

- Allowing other researchers to replicate your (computational) analysis of the data

- Reproducibility doesn't ensure correctness, but still helpful

- Not only required in bioinformatics research, but also required in statistical research

Why is RR important?

- Reduce (honest) mistakes
- Improve productivity for the long run
- More likely to be used, extended, and cited

General Principles of RR

- Keep track of how every result was produced:
  - Avoid manual data manipulation
  - Version control all custom code
  - Provide public access to data and code

One Principle:

Everything Automated with Code

Tools

1. Workflow automation: GNU make + CLI tools instead of GUI tools; Workflow & pipeline systems

2. R / Python packages instead of code snippets

3. knitr / IPython Notebook + Markdown instead of Word

4. Code version control: git / GitHub

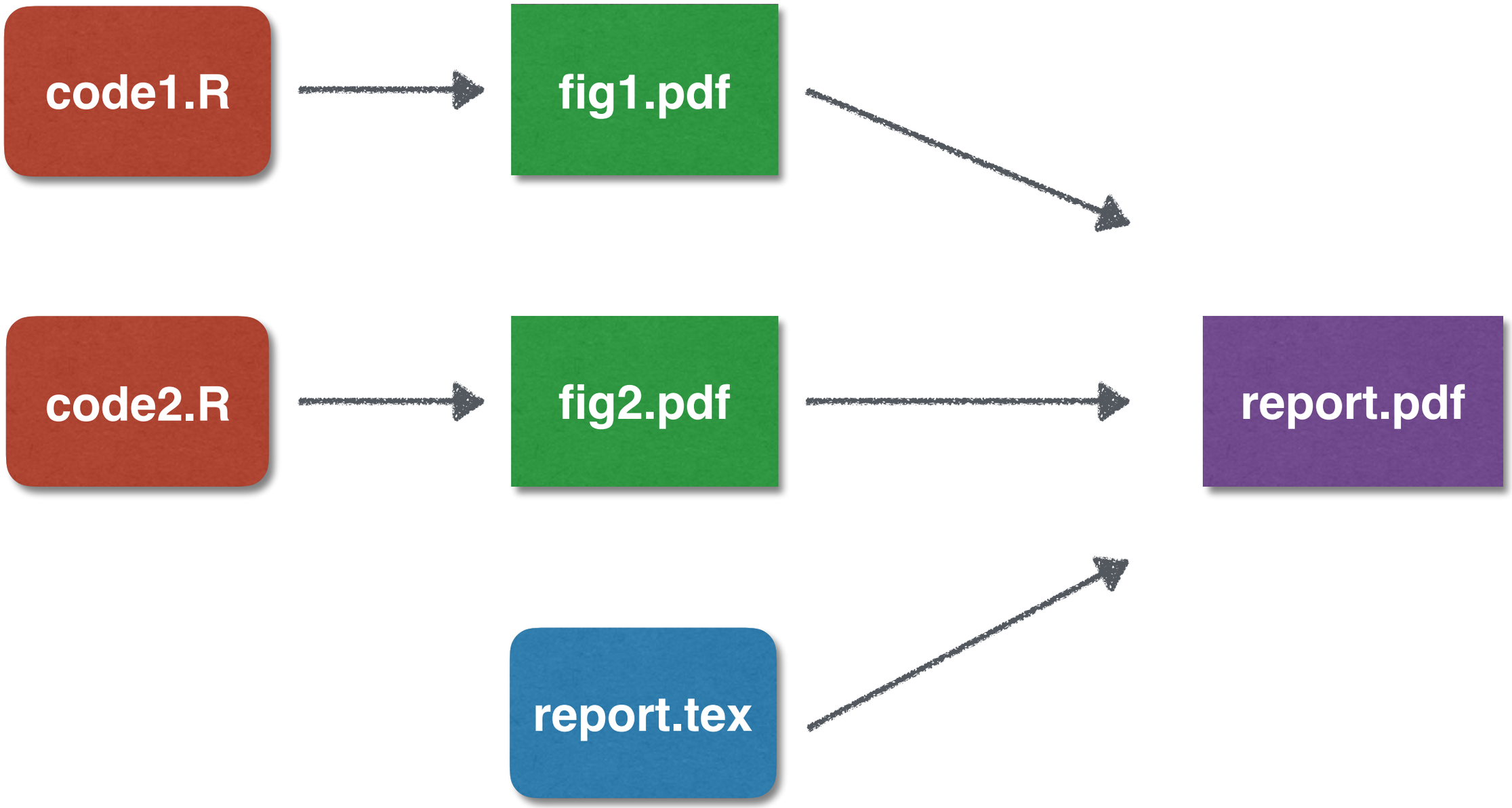
5. Package dependency management: packrat / virtualenv

6. System dependency management: Docker

1. Make & its friends

- Make can be used for computational project workflow automation
- Organises code & data dependencies naturally
- Works seamlessly with Linux/Unix CLI tools, such as sed, awk, and many others

Example Makefile

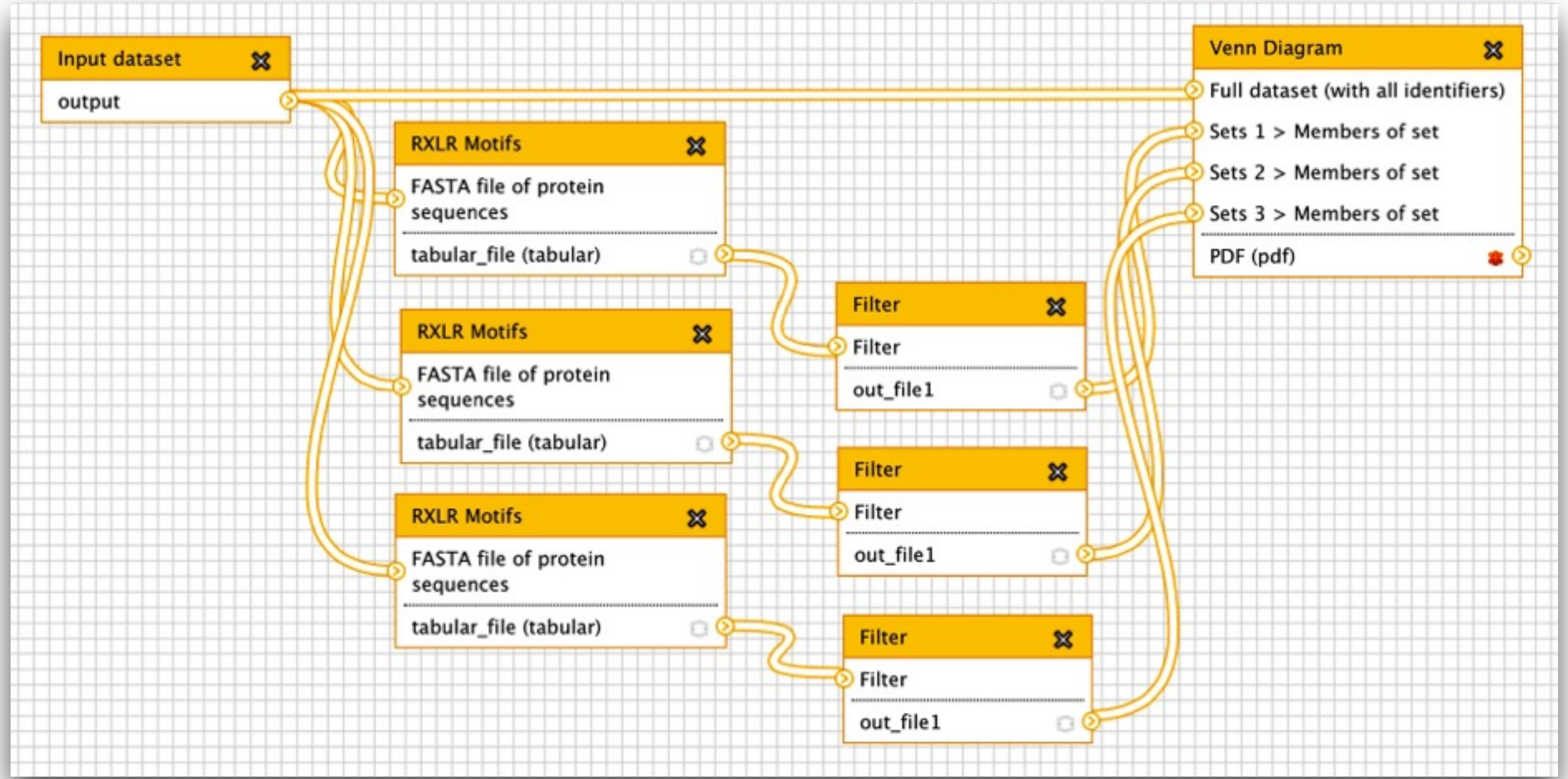


Example Makefile

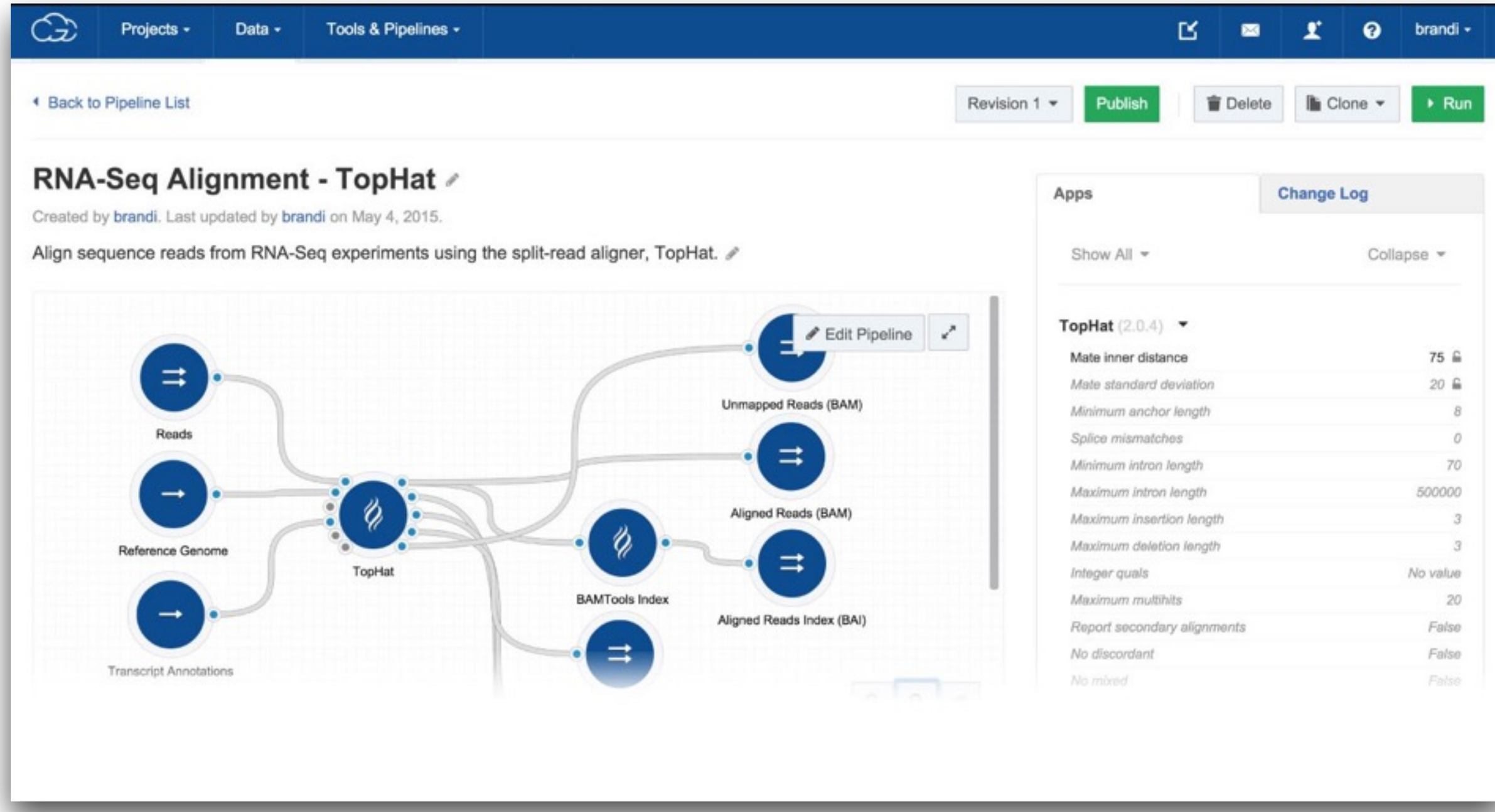
```
# Example Makefile for a LaTeX report

report.pdf: report.tex img/fig1.pdf img/fig2.pdf

pdflatex report


# Run R code to reproduce figures

img/fig1.pdf: Rcode/code1.R

cd Rcode; R CMD BATCH code1.R code1.Rout

img/fig2.pdf: Rcode/code2.R

cd Rcode; R CMD BATCH code2.R code2.Rout
```

Workflow Systems

- Galaxy
- bpipe
- systemPipeR
- Rabix
- … and many others:
- https://github.com/pditommaso/awesome-pipeline



Galaxy Workflow



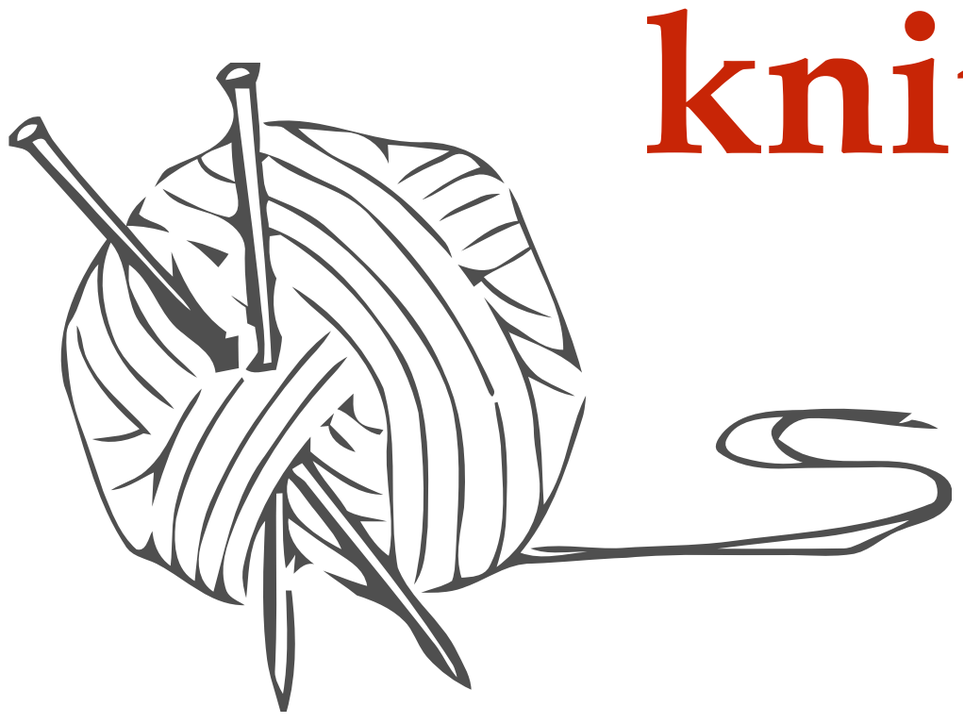
Pipeline in NCI Cancer Genomics Cloud
http://www.cancergenomicscloud.org

If you want to go beyond CLI and Galaxy …

R / Python packages can save the day.

2. R Packages

- readr: data loading
- httr: web scraping
- tidyr: data cleaning
- dplyr: data mangling
- stringr: string data
- lubridate: time data
- ggplot2: visualization
- devtools: package dev
- roxygen2: documentation
- testthat: unit testing



knitr

3. knitr

- knitr report = code + text

Let's explore our ChIP-seq's peak length:

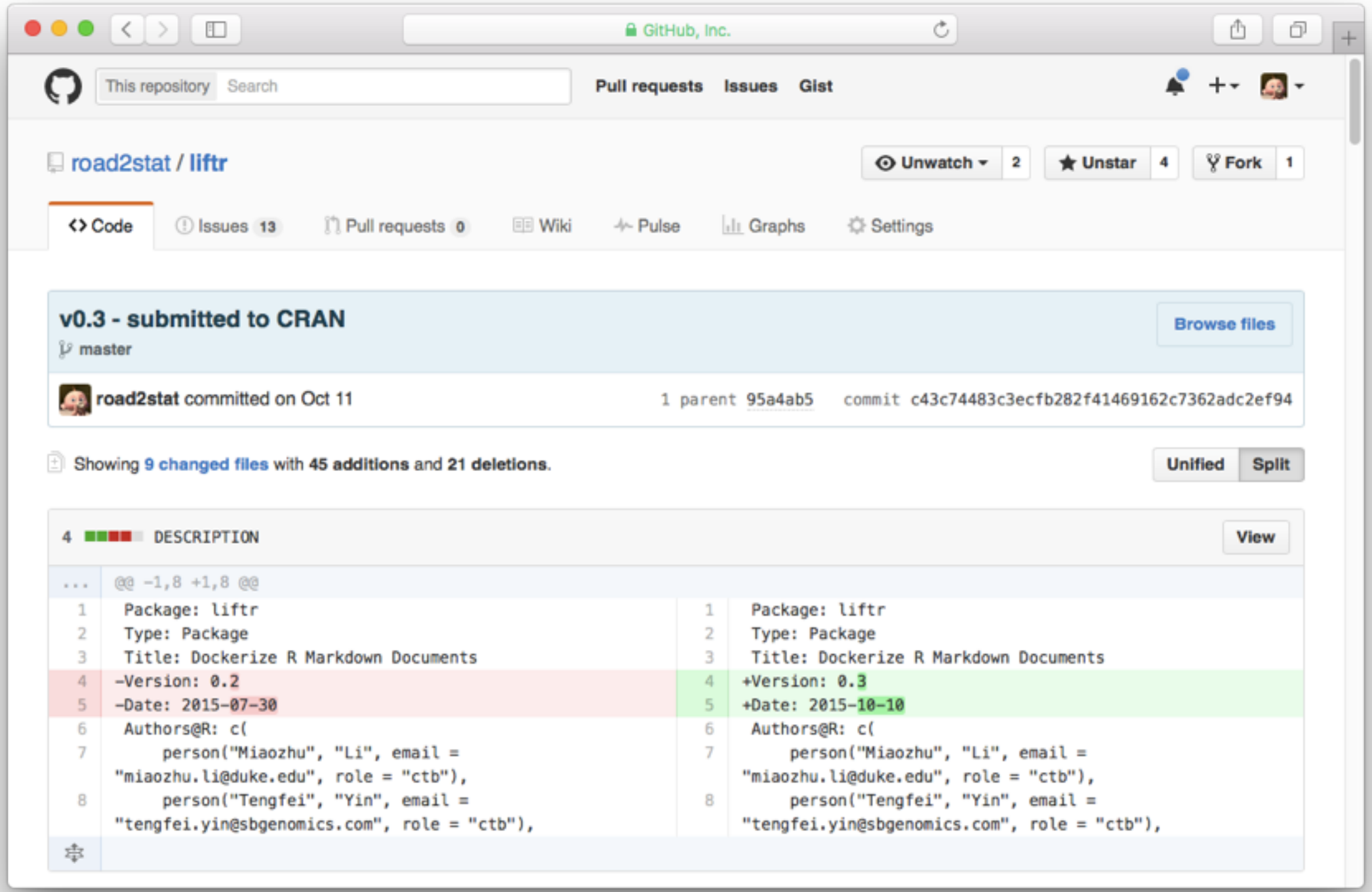
```{r}
counts = read.table("peaks.broadPeak")
hist(counts$end - counts$start)
```

The median of the peak length is
`r median(counts$end - counts$start)`.

Demo with RStudio

- RStudio
- knitr + RMarkdown
- Analyze ChIP-seq peak length distribution

4. Code Version Control



Easy to see when and where the code was changed

5. Package Dependency

- R: packrat
- Python: virtualenv

- Manages project package dependency, to make sure every newly created environment has the same versions of packages

6. System Dependency

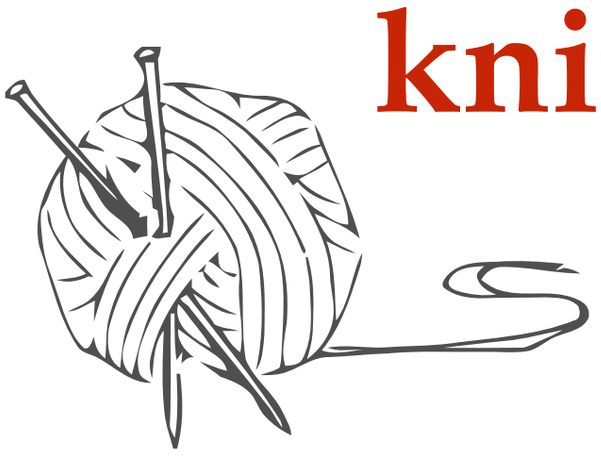
Challenges

- Allow for some level of interactivity: test & debug (knitr, IPython Notebook)
- Handling large-scale computation (Docker + AWS)
- Dependency deprecation (packrat)
- OS-level reproducibility (Docker)
- Data privacy concerns

liftr.me

Idea

- A framework for dockerizing R markdown documents
- Built-in Rabix (bioinformatics pipelines) support
- Reproducible bioinformatics and statistical analysis

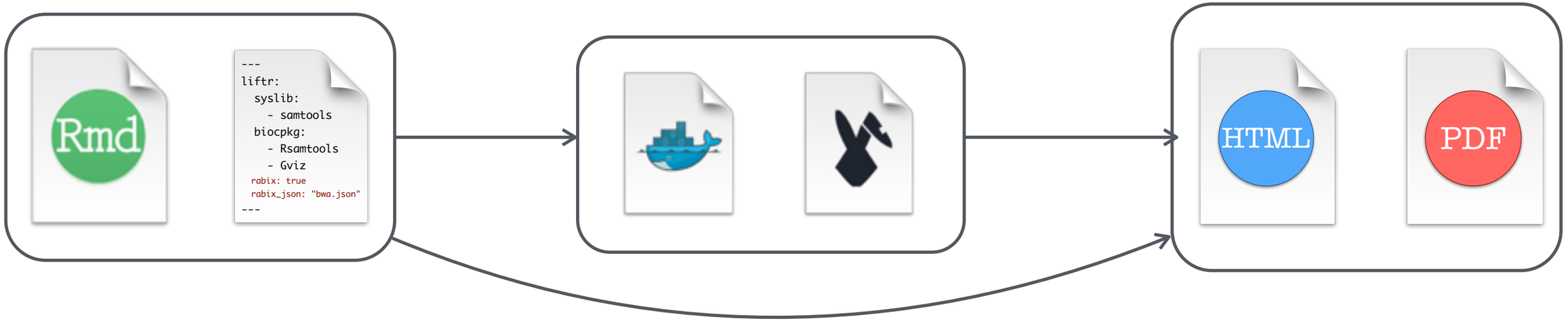

knitr


Rabix


docker

liftr workflow

`lift("foo.Rmd")` → `drender("foo.Rmd")` → Read and share!

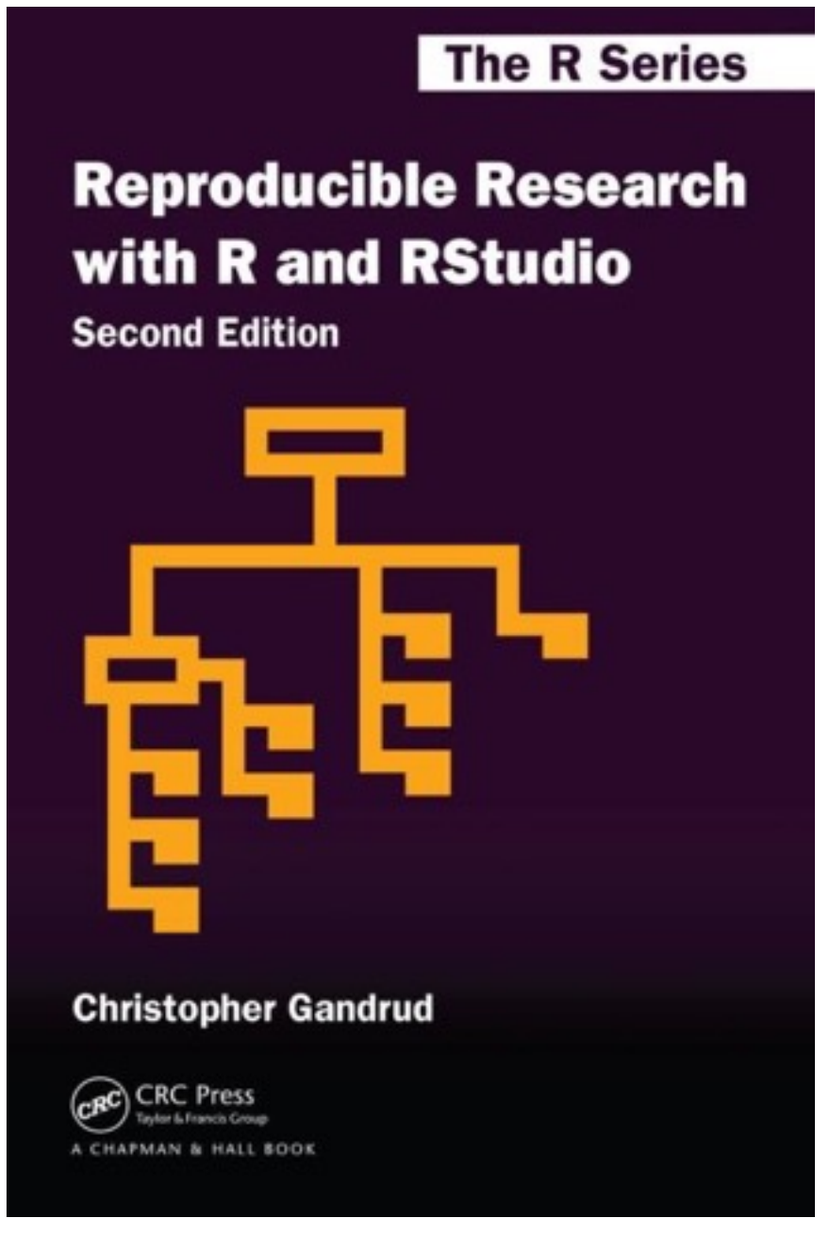


Rmd Documents with `` `liftr` `` options in metadata

Generated Dockerfile (Rabixfile)

Rendered HTML/ PDF/Docx Reports

Further Readings



Q & A

nanx.me
nanx@uchicago.edu