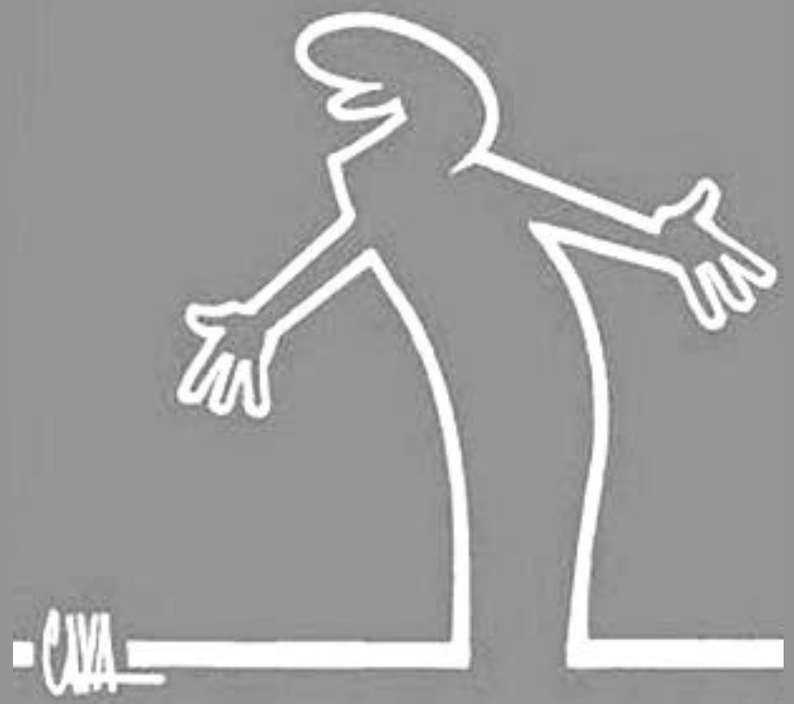
How to Structure a Data Analysis Project

Nico Scherf

- An overview on best (or good-enough) practices of how to organize data analysis projects.
    - Why ?
    - What ?
    - How ?
    - Resources

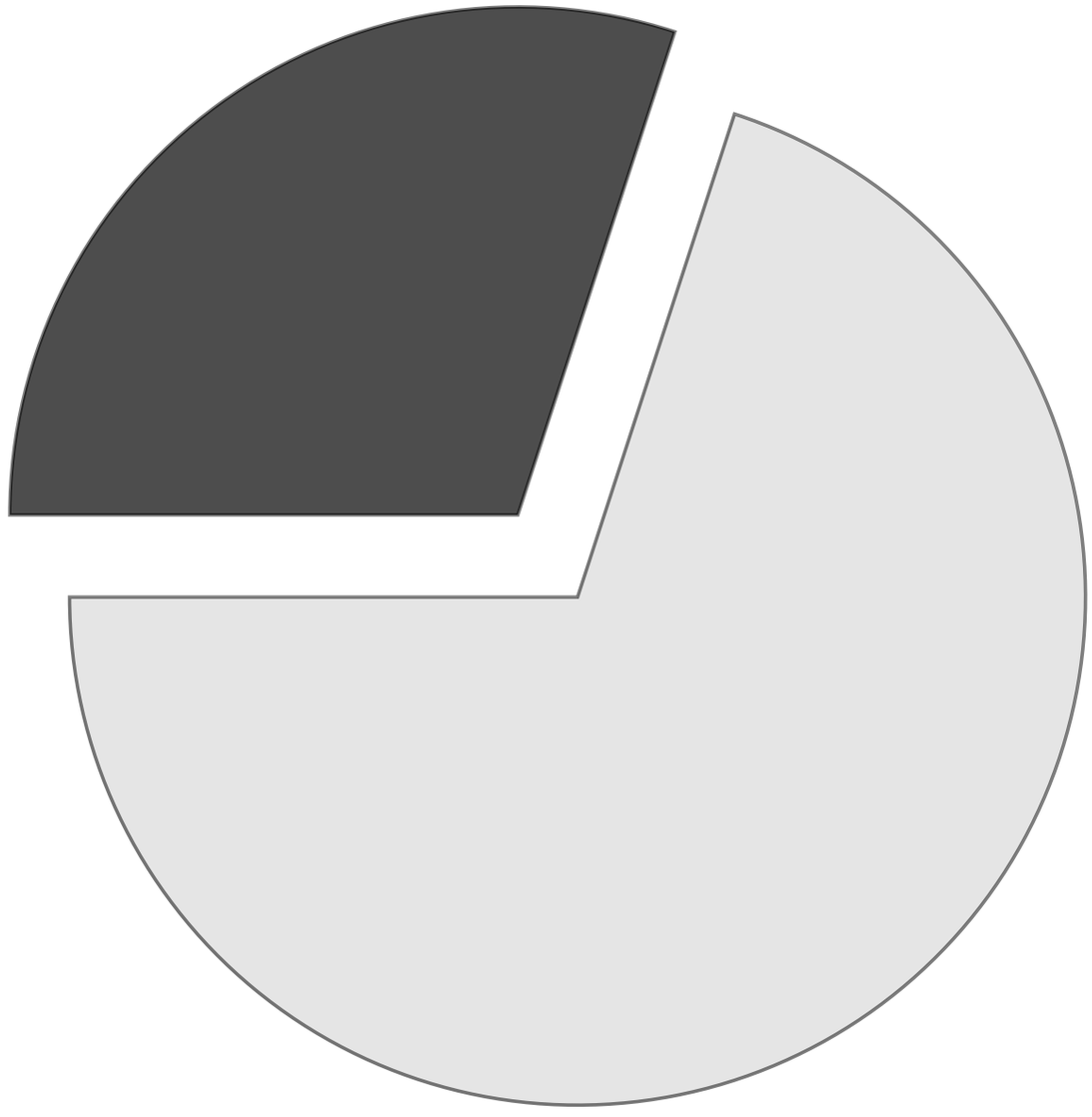


Why?

Computational Data Analysis in Research

- Software is as important to modern research as telescopes and tubes

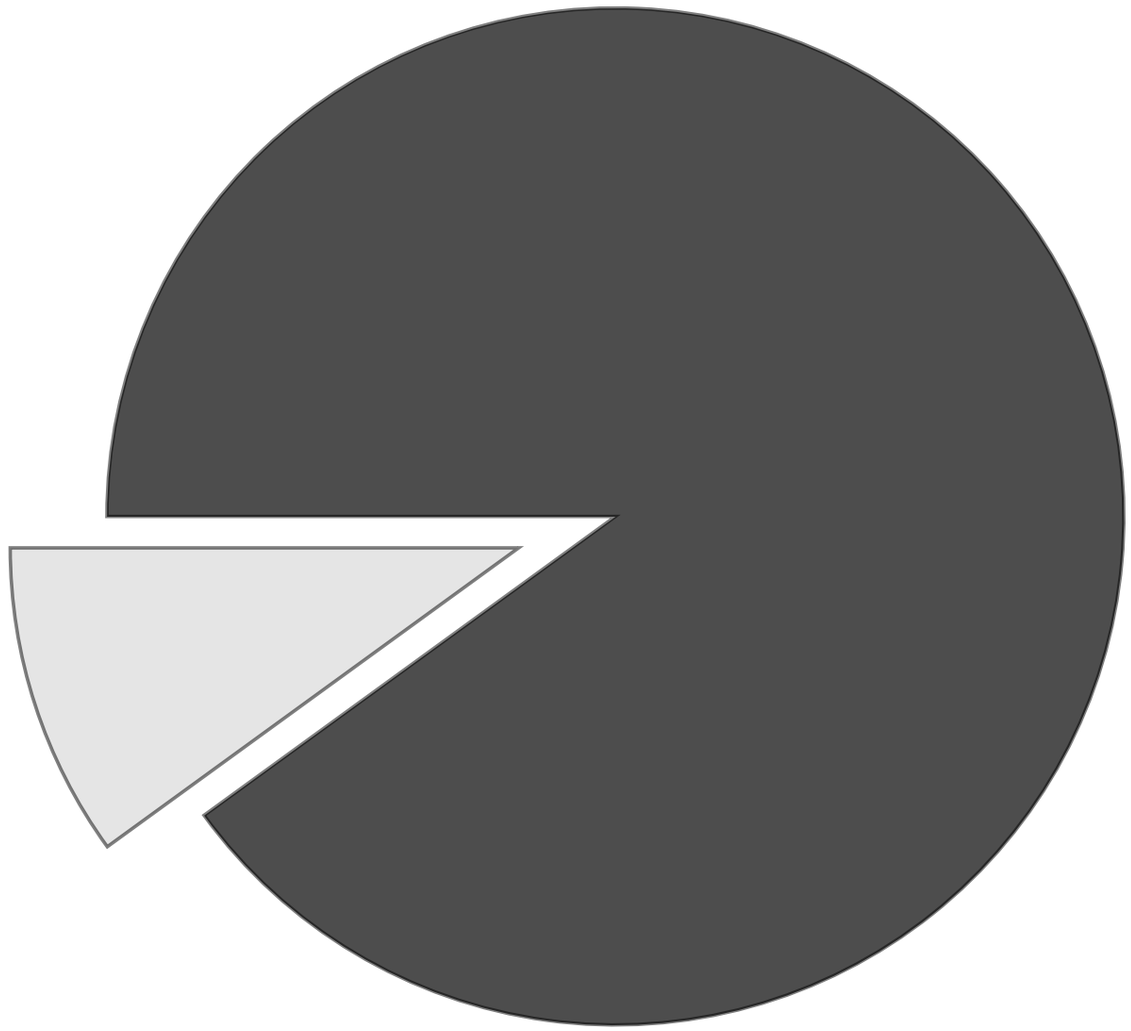
- Software is just another kind of experimental apparatus and should be built, checked and used as carefully as any physical apparatus



- Scientists spend 30% of time developing software

Hannay, Jo Erskine, Carolyn MacLeod, Janice Singer, Hans Petter Langtangen, Dietmar Pfahl, and Greg Wilson. 2009. “How Do Scientists Develop and Use Scientific Software?” In *Proceedings of the 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering*, 1–8. SECSE ’09. Washington, DC, USA: IEEE Computer Society.

- >90% are self-taught in programming



Hannay, Jo Erskine, Carolyn MacLeod, Janice Singer, Hans Petter Langtangen, Dietmar Pfahl, and Greg Wilson. 2009. “How Do Scientists Develop and Use Scientific Software?” In *Proceedings of the 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering*, 1–8. SECSE ’09. Washington, DC, USA: IEEE Computer Society.

- Not everything must be done to most exacting standards
- Be aware of best practices
- And think about your future self !

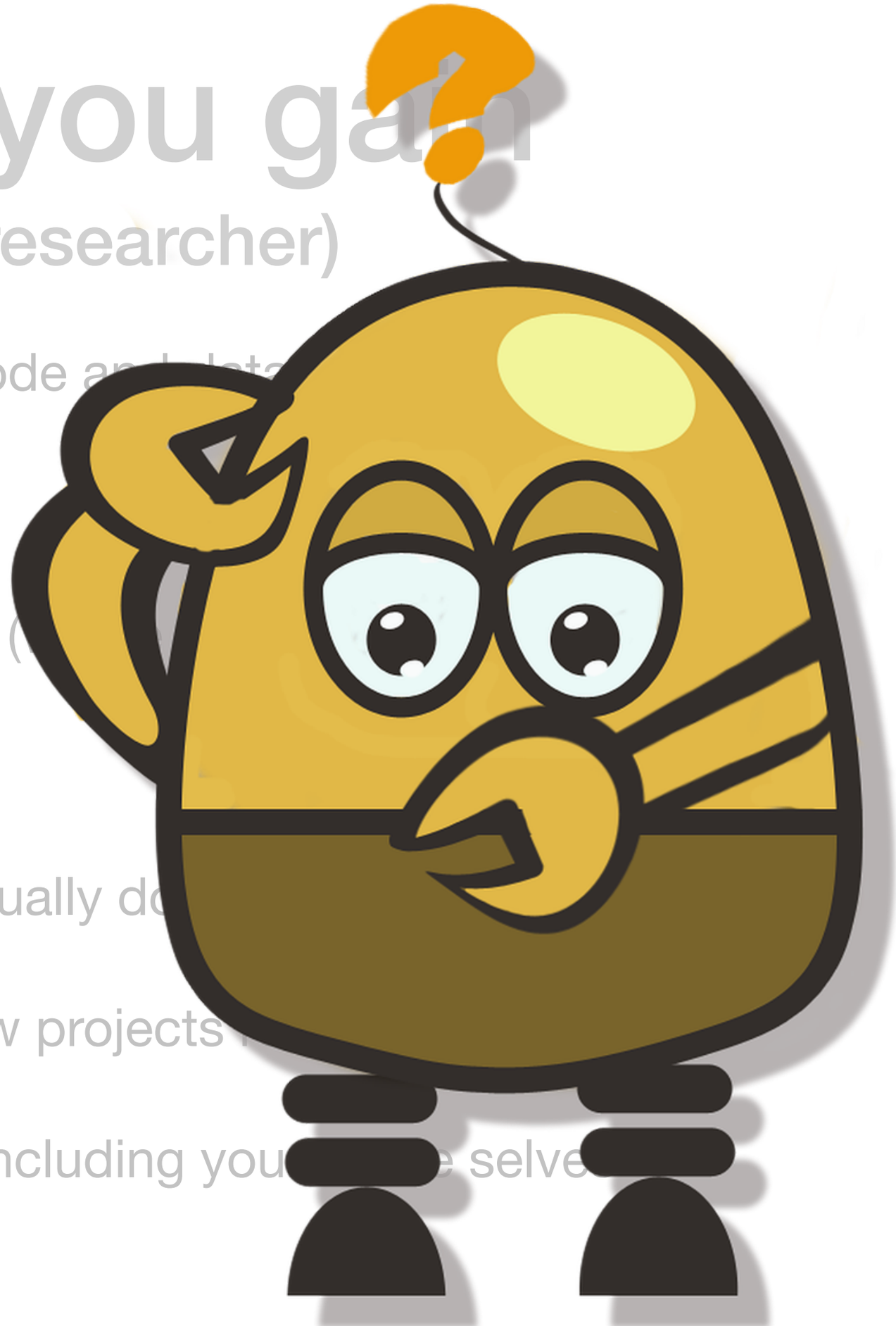
What you gain
(as a researcher)

- Efficient and convenient way to share code and data
- simplified file management (where did I put that?)
- streamlined workflows
    - reproducibility, you can actually do it again...
    - startup and re-entry for new projects reduced
- communicate with others including your future selves

What you gain

(as a researcher)

- convenient way to share code an
- increased efficiency
- simplified file management (
- streamlined workflows
- reproducibility, you can actually d
- startup and re-entry for new projects
- communicate with others including you



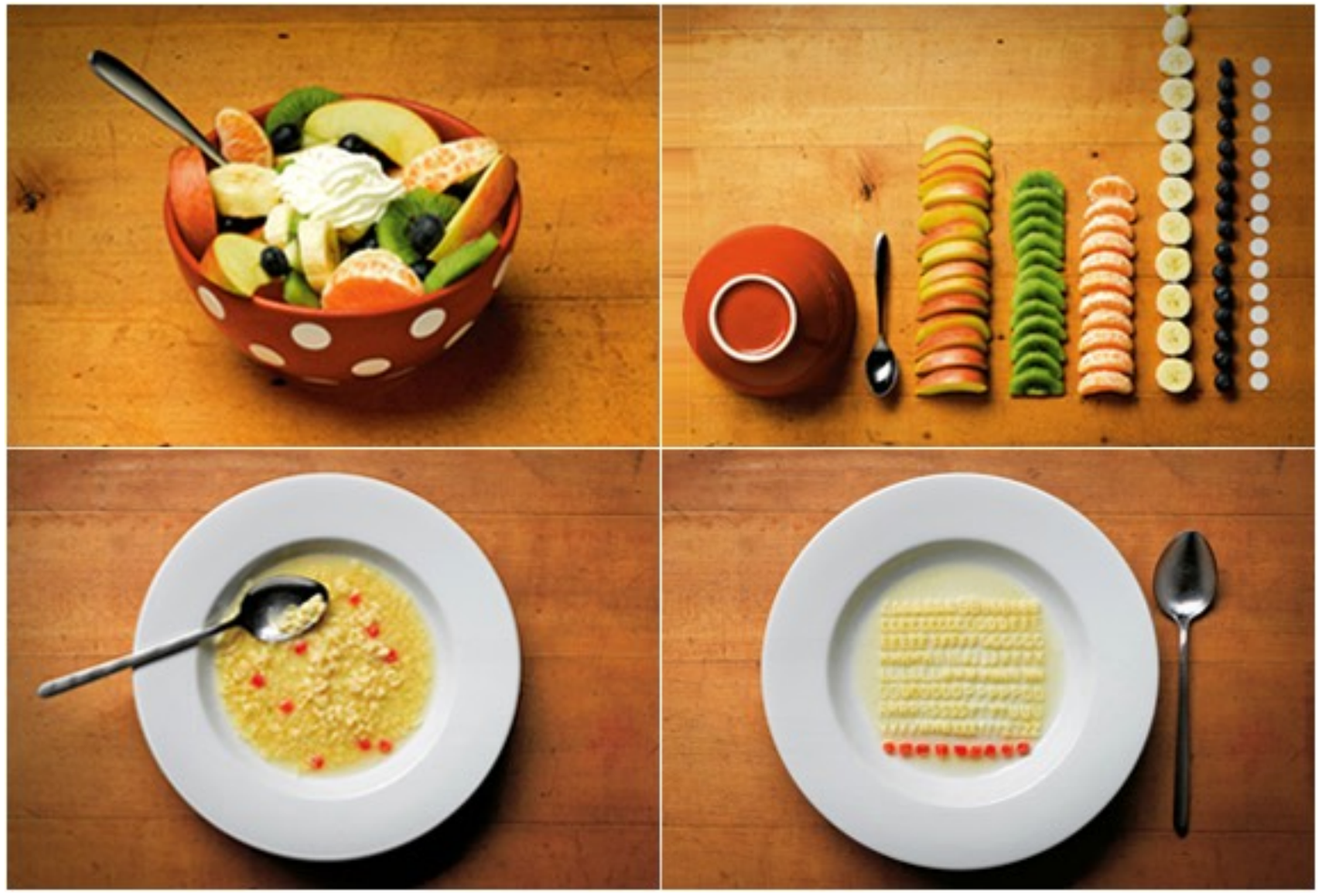
What?

A research compendium

Packaging data analytical work reproducibly
using R (and friends) - Marwick, Boettiger, Mullen
https://peerj.com/preprints/3192.pdf

A research compendium

- The goal of a research compendium is to provide a standard and easily recognisable way for organising the digital materials of a project to enable others to inspect, reproduce, and extend the research.



Ursus Wehrli - The Art of Tidying Up

Reproducible Research

- reproducing research is the calculation of quantitative scientific results by independent researchers using the original data and methods

Reproducible Research

- Empirical (reagents, cell lines, sample identities, instrument settings)
- Statistical (details of tests, model parameters…)
- Computational (details about code, software, hardware and implementation)

Stodden (2014) WHAT SCIENTIFIC IDEA IS READY FOR RETIREMENT? Reproducibility https://www.edge.org/response-detail/25340

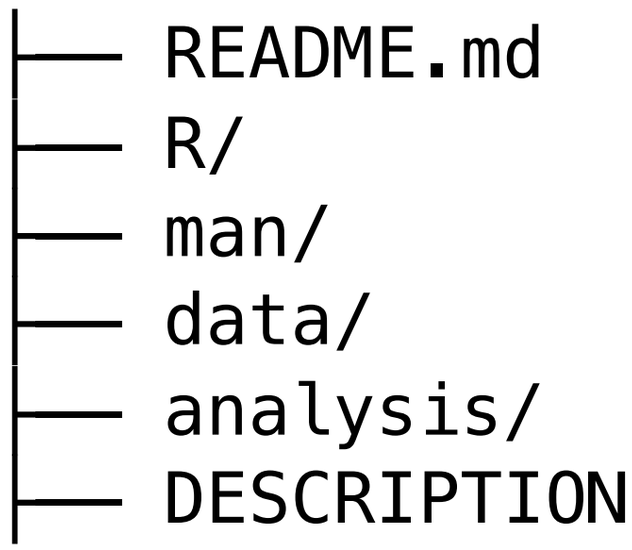
Generic Principles

1. Organizing files according to prevailing conventions of community

2. Clear separation of data, method, and output

3. Specify computational environment used for original analysis

Generic Principles



- Organizing files according to prevailing conventions of community helps other to read structure of project

Generic Principles

- Clear separation of data, method (code), and output

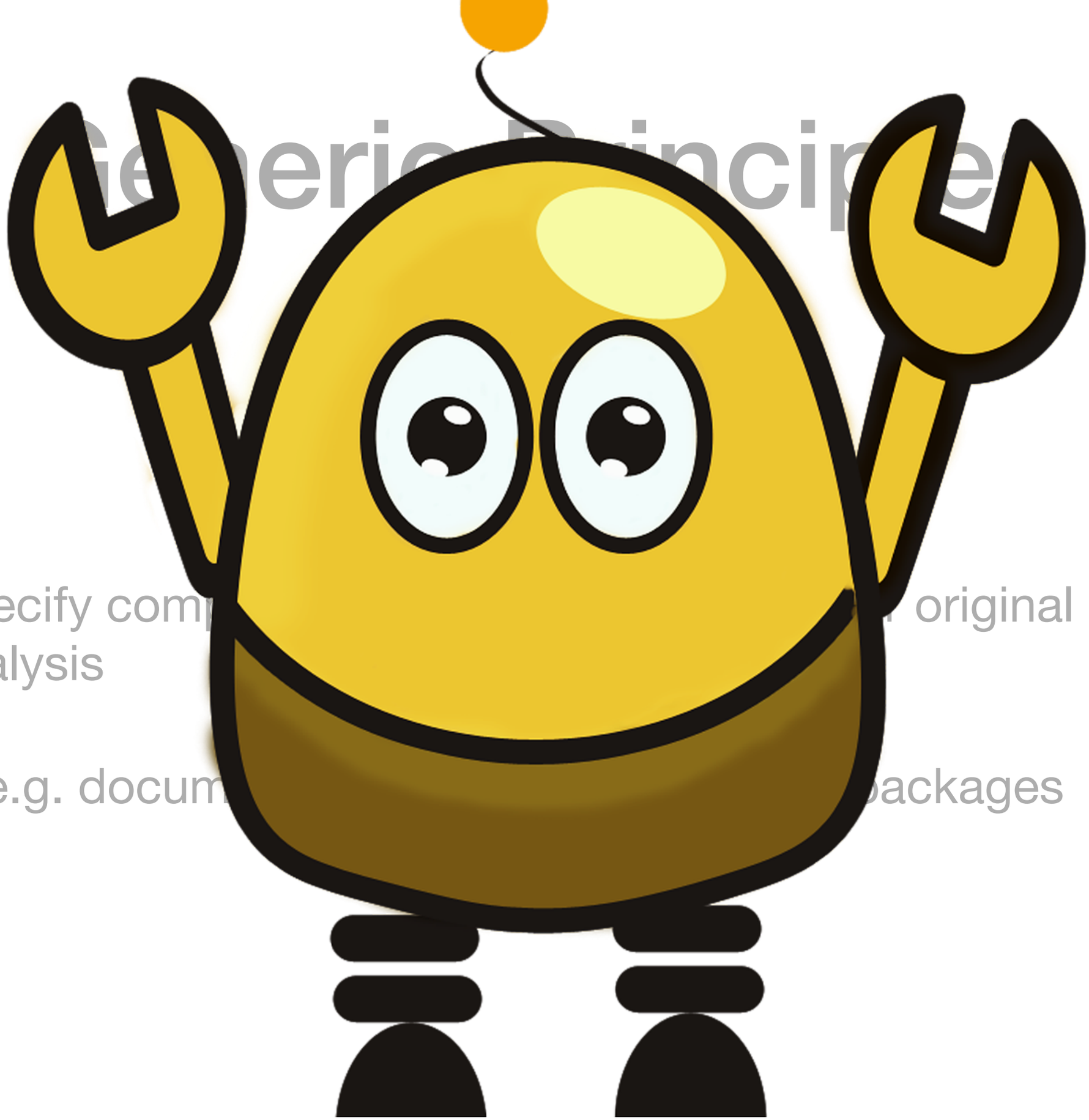
- data is read only, modifications are documented in code

- output files are disposable, can be regenerated using code and data

Generic Principles

- Specify computational environment used for original analysis
    - e.g. document versions of libraries and packages

Generic Principles



- Specify comp… original analysis
  - e.g. docum… packages

Related

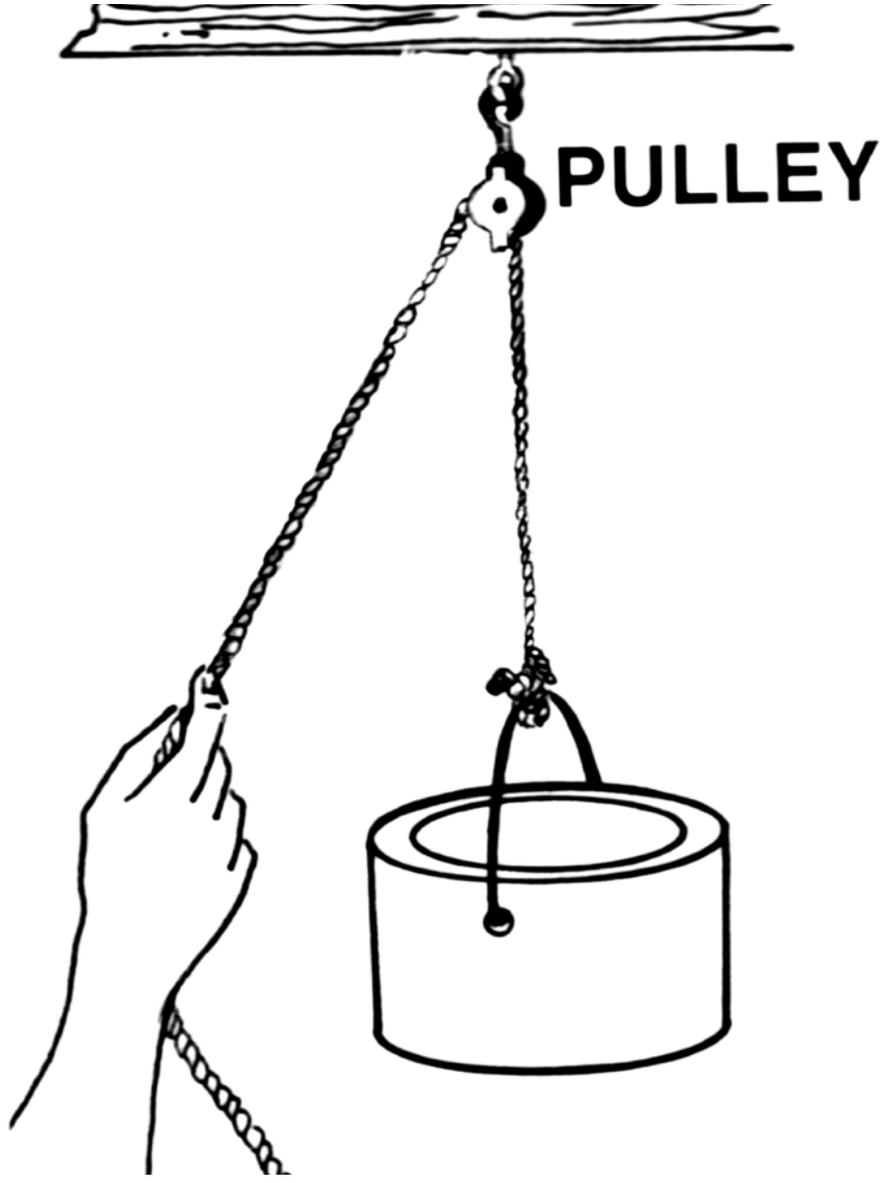
- Use the R package structure as a basic layout
- writing R package enforces quality control

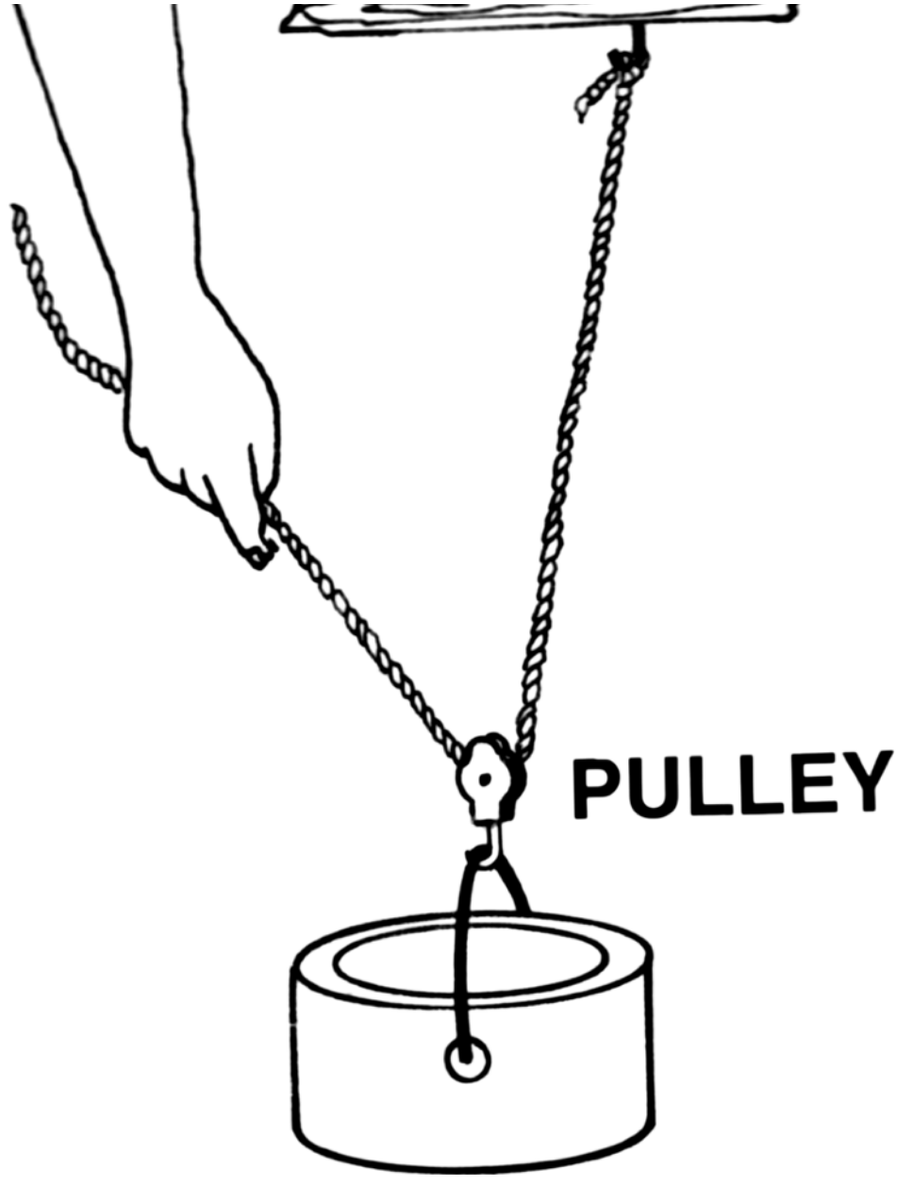
Best practices for Scientific Computing

- Write programs for people, not computers
- Let the computer do the work
- Make incremental changes
- Don’t repeat yourself
- Plan for mistakes
- Optimize software only after it works correctly
- Document design and purpose, not mechanics

Wilson, Greg, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, et al. 2014. “Best Practices for Scientific Computing.” PLoS Biology 12 (1): e1001745.

How?





The simple way

Generic Principles

- Clear separation of data, method, and output
  - data :"read only", modifications are documented in code
  - output files are disposable, can be regenerated using code and data

- A single R file with code and inline documentation

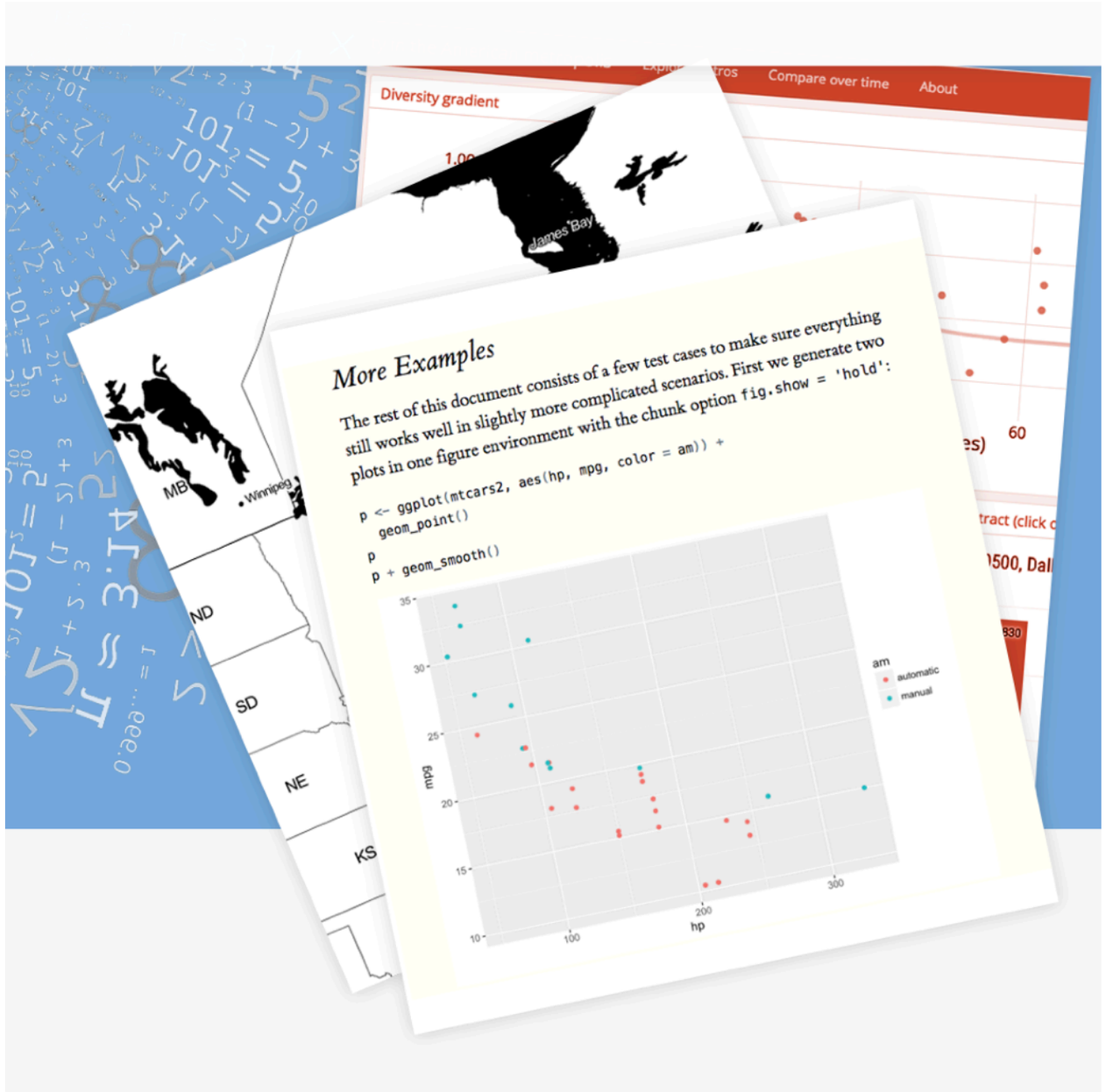
- Accompanying data files

309 lines (241 sloc) | 13 KB

```
#=============================================================================
# Script created by Meghan Duffy, duffymeg@umich.edu
# Script created in version R 3.1.2
# This script is for analyzing data related to the Duffy et al. paper re-desc
# the Daphnia brood parasite Blastulidium paedophthorum (Bp)
#=============================================================================

# Set working directory
#use line below if working on pc
# setwd("C:/Users/duffymeg/Box Sync/Parasites/BroodParasite/CodeforDuffyetalB
#use line below if working on mac


# load all data
# North Fall 2013 life table
brooddataNorth2013 <- read.table("data/NorthBPLifeTable2013.txt", header=TRUE
# Now moving on to Fall 2014 data
# North Lake:
brooddataNorth2014 <- read.csv("data/NorthBPLifeTable2014.csv", header=TRUE,
brooddataNorth2014dentifera <- subset(brooddataNorth2014,Treatment=="Infected
brooddataNorth2014retrocurva <- subset(brooddataNorth2014,Treatment=="Infecte
# Cedar Lake:
brooddataCedar2014 <- read.csv("data/CedarBPLifeTable2014.csv", header=TRUE,
brooddataCedar2014dentifera <- subset(brooddataCedar2014,Treatment=="Infected
brooddataCedar2014retrocurva <- subset(brooddataCedar2014,Treatment=="Infecte

# Survival analyses
library(survival)
#Note "Censored" column is whether or not the animal died before experimental

# North 2013 data:
model1 <- survfit(Surv(brooddataNorth2013$Day.Death, brooddataNorth2013$Censo
plot(model1, lty=c(5,1),lwd=3)

tapply(brooddataNorth2013$Day.Death,brooddataNorth2013$Treatment,mean,na.rm=T
```

https://github.com/duffymeg/BroodParasiteDescription

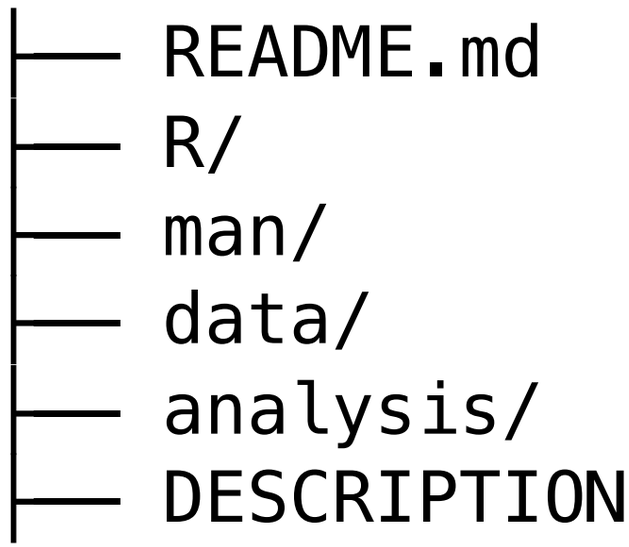
- Even better:
- R markdown file using text and code
  - Literate programming style
- Accompanying data files



Generic Principles

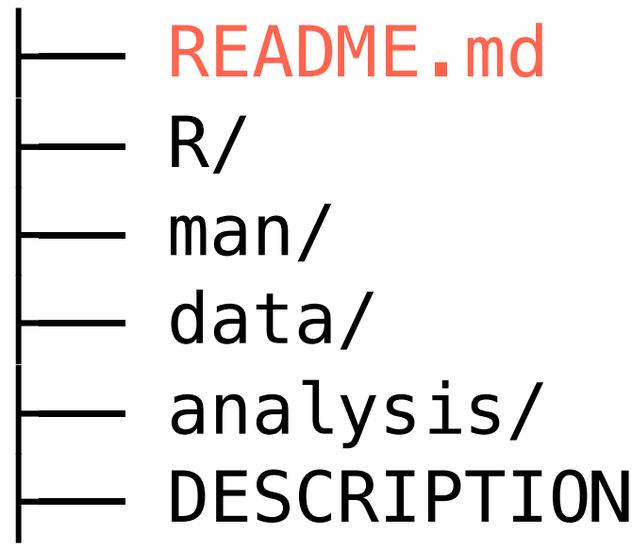
- Organizing files according to prevailing conventions of community
    - helps other to read structure of project

Use R package conventions



Use R package conventions

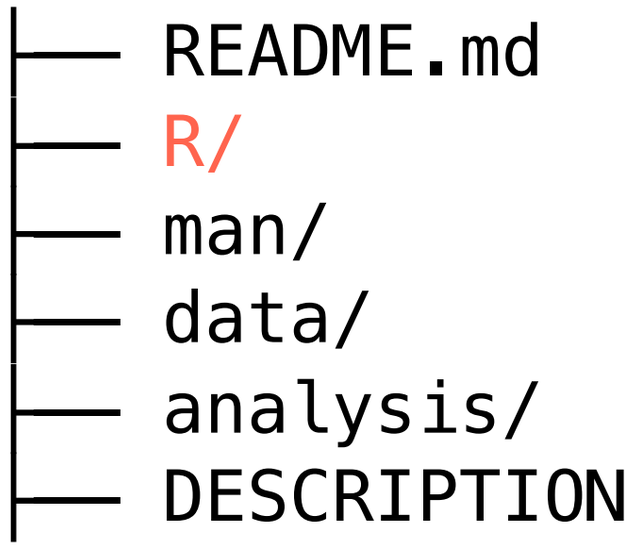
- README.md: describing overall project and where to get started



Use R package conventions

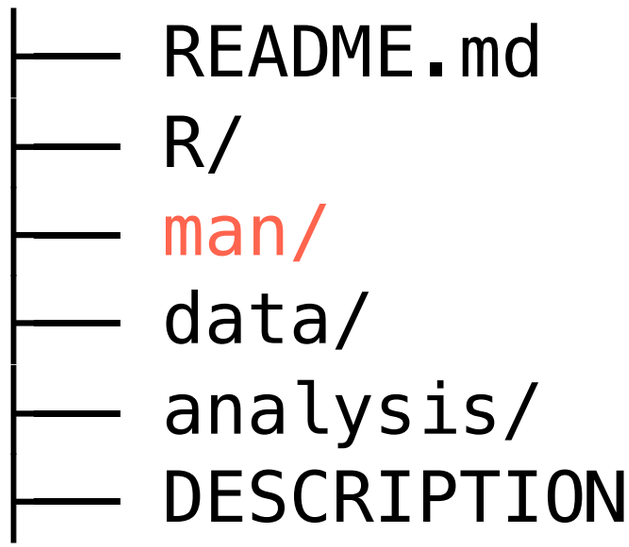
- R/ : script files with (reusable) functions,

- if you use roxygen, documentation will be generated in



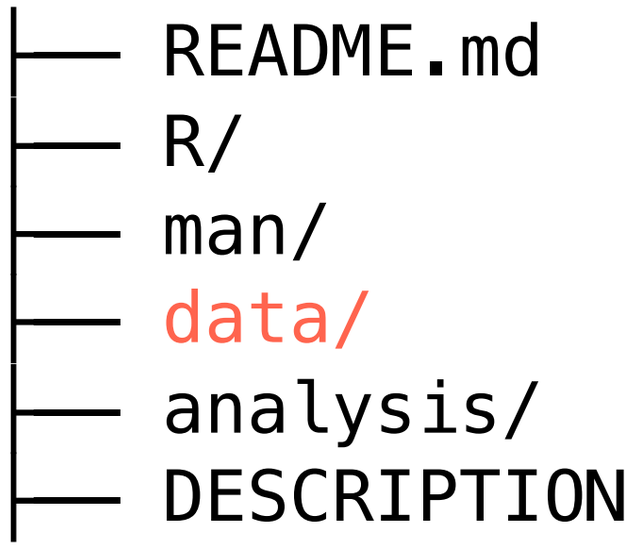
Use R package conventions

- man/
  - Documentation goes here…



Use R package conventions

- data/ raw data files
- a small sample set if actual data are large



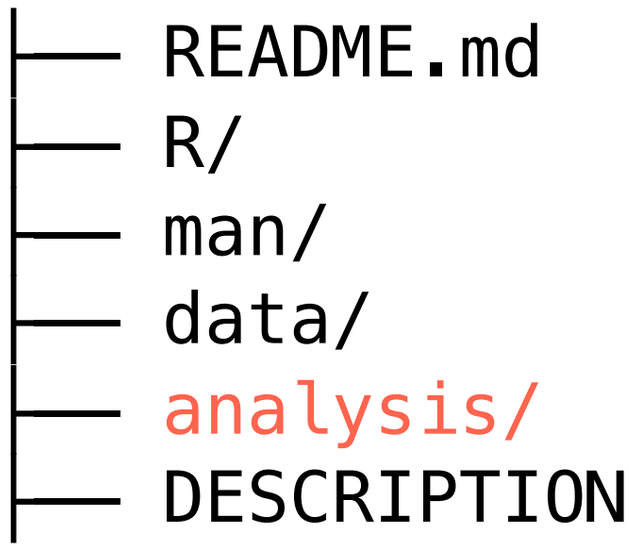
Use R package conventions

- analysis/ scripts for analysis and reports files

  - use ascending names: 001-load.R, 002-clean.R

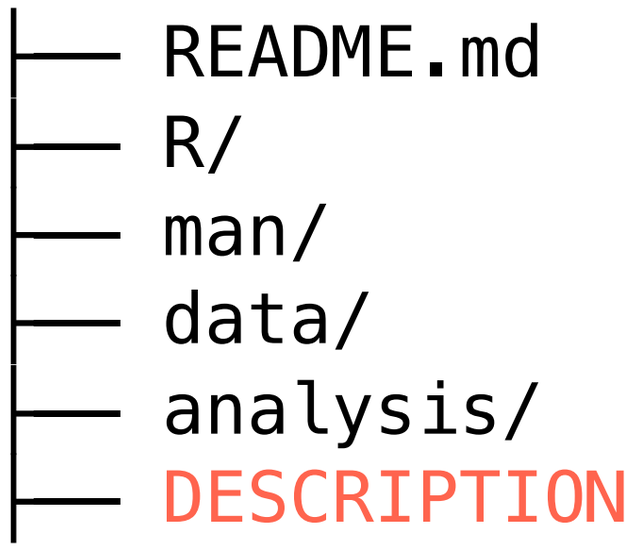
  - include a Makefile or a R Markdown file for complex workflows

    - control (and document) order of code execution



Use R package conventions

- DESCRIPTION machine- and human-readable information about authors, license, dependencies, and metadata



Use R package conventions

- if you use all of that, you get an installable R package!
    - you can benefit from testing, sharing, citation() function etc...

```
├── README.md
├── R/
├── man/
├── data/
├── analysis/
└── DESCRIPTION
```

Best practices

- Write programs for people, not computers
- Let the computer do the work
- Make incremental changes
- Don’t repeat yourself

Wilson, Greg, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, et al. 2014. “Best Practices for Scientific Computing.” PLoS Biology 12 (1): e1001745.

Best practices

- Write programs for people, not computers
- Write code that others can read and understand
  - Including your future self 

Best practices

- Write programs for people, not computers
- Code should not require to hold more than a handful of facts in memory -> use functions

- Alan Perlis: “It is better to have 100 functions operate on one data structure than 10 functions on 10 data structures.”

Best practices

- Write programs for people, not computers
- Make names consistent and meaningful

Best practices

$data

- Write programs for people, not computers
- Make names consistent and meaningful

http://archive.oreilly.com/pub/post/the_worlds_two_worst_variable.html

Best practices

$data2

- Write programs for people, not computers
- Make names consistent and meaningful

http://archive.oreilly.com/pub/post/the_worlds_two_worst_variable.html

Best practices

```
$total = $price * $qty;

$total2 = $total - $discount;

$total2 += $total * $taxrate;
```

```
$total3 = $purchase_order_value + $available_credit;

if ( $total2 < $total3 ) {

    print "You can't afford this order.";

}
```

http://archive.oreilly.com/pub/post/the_worlds_two_worst_variable.html

Best practices

- Let the computer do the work
- Make computer repeat tasks
- Write scripts
- Use build tools for more complex workflows

Best practices

- Let the computer do the work
    - Unique identifiers and version numbers for
        - raw data
        - Programs and libraries
- Values of parameters

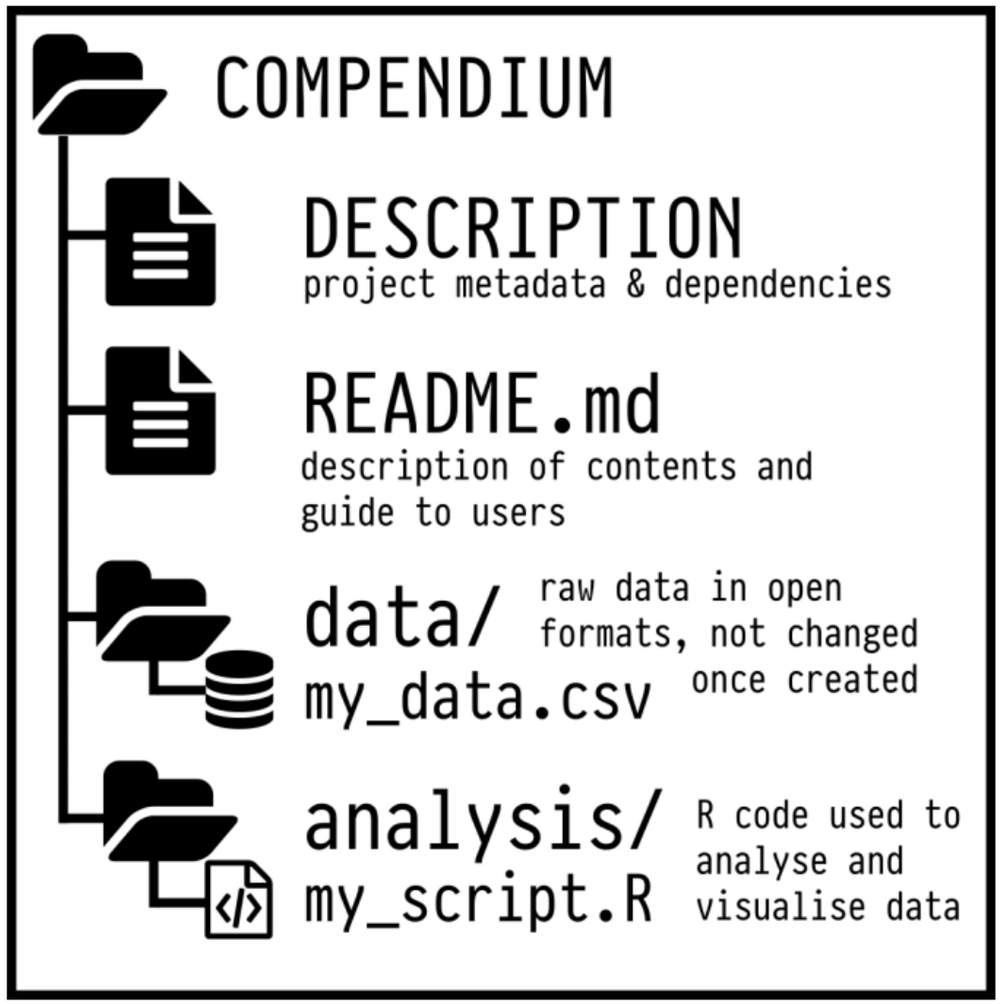
Best practices

- Make incremental changes
    - Work in small steps with frequent feedback and course correction
- Use version control
    - Put everything that has been created manually under version control

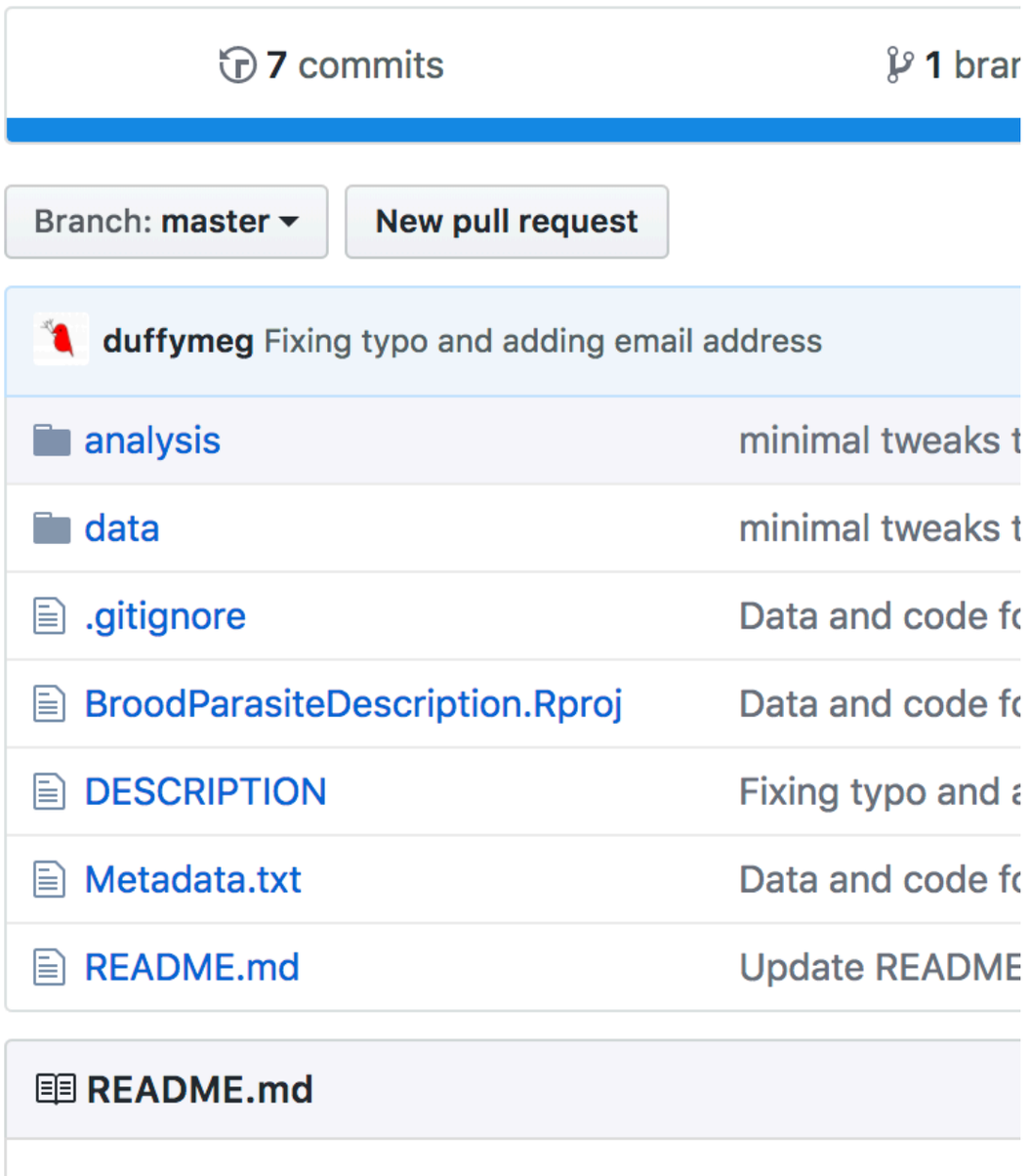
Examples of real-world research compendia using R packages

A small one

- Duffy 2015 - parasite study
  - http://dx.doi.org/10.5281/zenodo.17804



- Uses convention for file organization
- Separates data and code
- Specifies computational environment



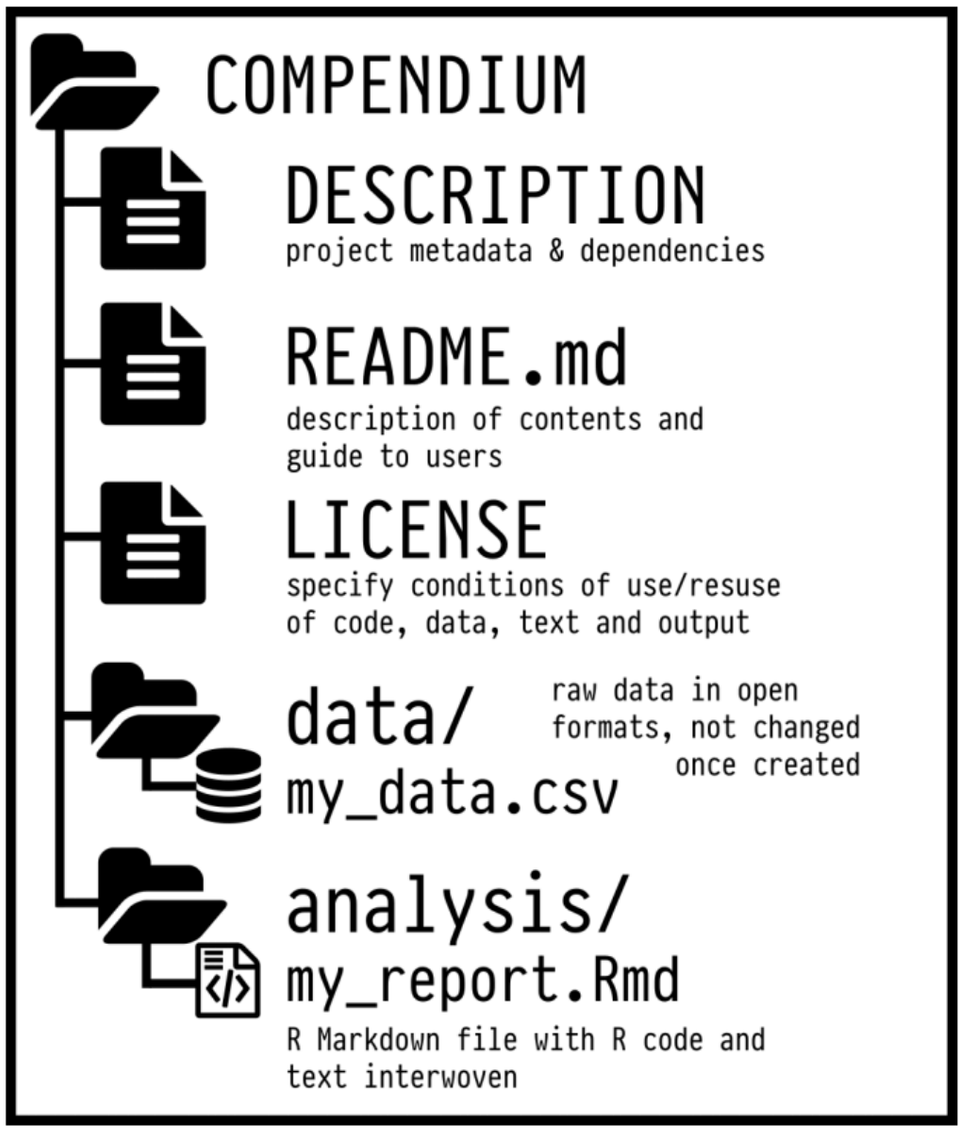
https: //github.com/duffymeg/BroodParasiteDescription

- Specifies computational environment in DESCRIPTION

17 lines (16 sloc) | 591 Bytes

```
1  Package: BroodParasiteDescription
2  Title: Brood Parasite Description
3  Version: 0.0.0.9000
4  Authors@R: person("Meghan", "Duffy",
5  Description: Code and data associate
6  Depends: R (>= 3.2.0),
7      survival,
8      stringr,
9      plyr,
10     ggplot2,
11     coin,
12     reshape2,
13     scales,
14     dplyr,
15     grid
16 License: Unknown
```

- A bit more complex:
  - incl. markdown
    - creates data visualisations and statistical contents of publication + basic comments
- LICENSE



R code for analysis of the Spirit Cave faunal data.

Cyler Conrad, Department of Anthropology, University of New Mexico, cylerc@unm.edu

Ben Marwick, Department of Anthropology, University of Washington, bmarwick@uw.edu

This document contain R code to reproduce the plots and statistical analysis presented in

> Conrad, C., Higham, C., Eda, M. and Marwick, B. (in press) Paleoecology and Forager Subsistence Strategies During the Pleistocene-Holocene Transition: A Reinvestigation of the Zooarchaeological Assemblage from Spirit Cave, Mae Hong Son Province, Thailand. Asian Perspectives.

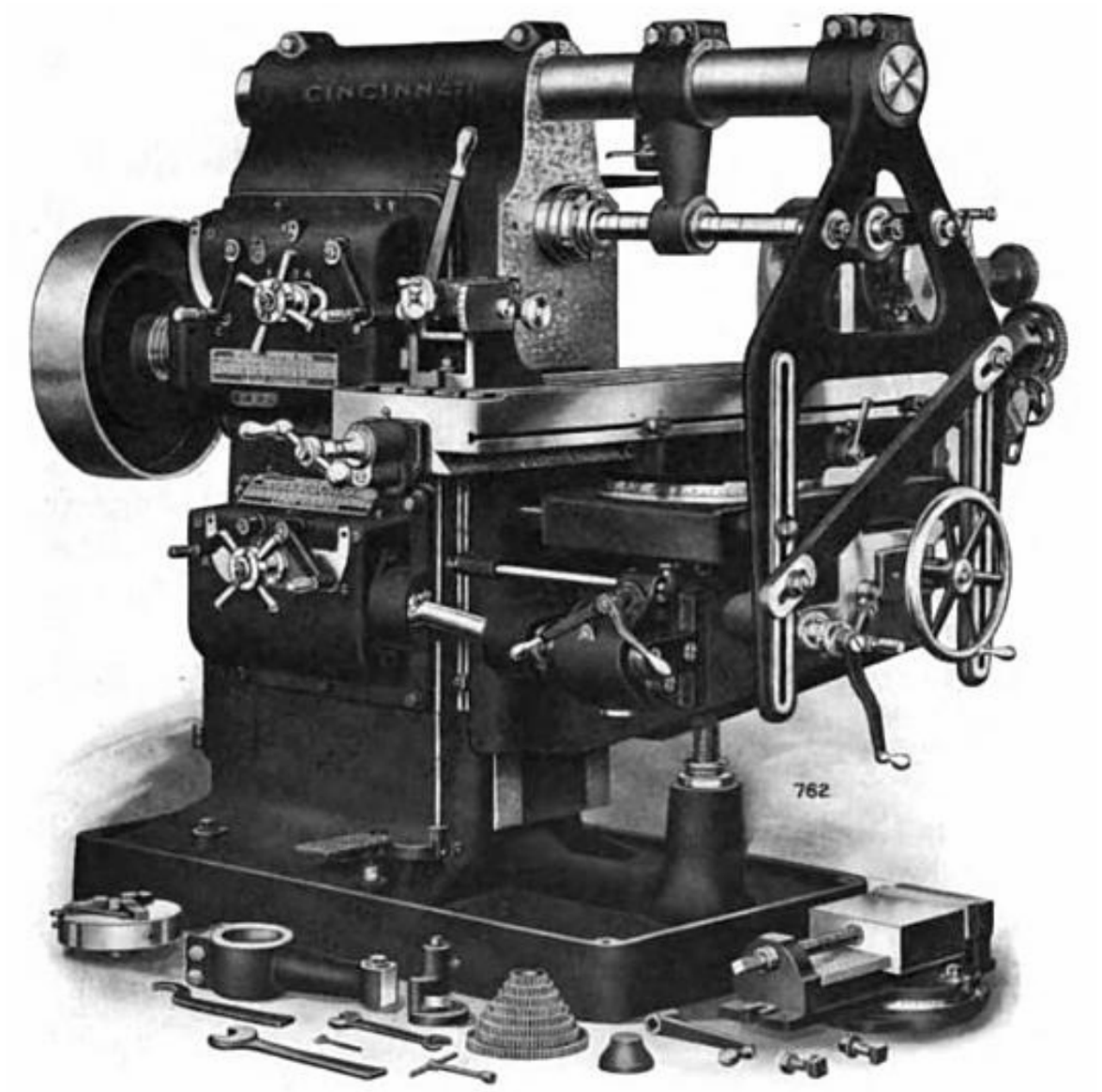
All data required to perform the analyses can be found at the University of New Mexico digital electronic repository (Conrad 2015). The development version of this document can be found at https://github.com/cylerc/AP_SC

Details of the computational environment and software dependencies are listed at the end of this document.

References: Conrad, C. 2015. Archaeological Databases for Spirit Cave, Mae Hong Son Province, Thailand [dataset]. University of New Mexico. http://repository.unm.edu/handle/1928/26730

```
# set the base directory for knitr to the directory above this one
# install.packages(c("knitr", "Bchron", "car", "ggplot2", "reshape2", "dplyr"))
library(knitr)
opts_knit$set(root.dir = '../', progress = FALSE)
```

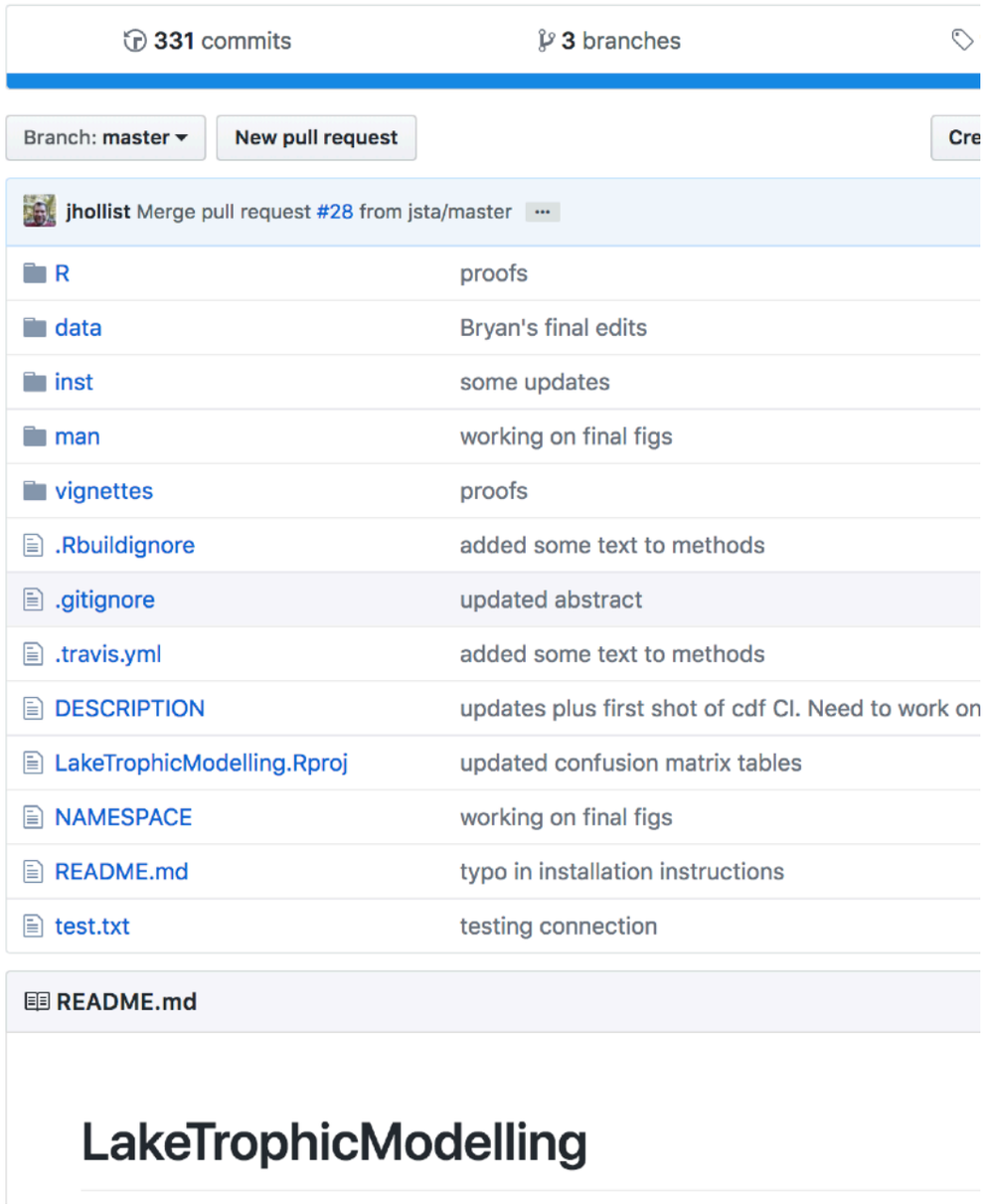
```
# see the output of sessionInfo() at the bottom for package version numbers
library(Bchron)
library(car)
library(ggplot2)
library(reshape2)
library(plyr)
library(dplyr)
library(viridis)
```



Examples of real-world research compendia using R packages

A medium compendium

- include
- R/ for custom functions
- man/ how to use functions
- NAMESPACE exports functions in a package



https://github.com/USEPA/LakeTrophicModelling

- include
- R/ for custom functions
- man/ how to use functions
- NAMESPACE exports functions in a package

Branch: master ▾ LakeTrophicModelling / R /

jhollist proofs

..	
LakeTrophicModelling-package.r	added some text to m
class_prob_rf.R	re rendered
condAccuracy.R	closes #24
crossval_rf.R	new udpates
density_plot.R	draft to send to B&B
ecdf_ks_ci.R	almost done with resu
ecor_map.R	closes #24
getCyanoAbund.R	fixed bum link to data
getLakeIDs.R	add getlakes some mo
importancePlot.R	some figure tweaks
iterVarSelRF.R	added some text to m
multiplot.R	added some text to m
nlaMap.R	closes #24
partial_plot.R	proofs
plotCdf.R	submission almost rea

https://github.com/USEPA/LakeTrophicModelling

- include
- R/ for custom functions
- man/ how to use functions
- NAMESPACE exports functions in a package

44 lines (44 sloc) | 2.1 KB

```
#' Get Class Prediction Probabilities
#'
#' This function takes a regression randomForest and returns class pr
#' for a provided classifcation.  These probabilities are based on a
#' If using the dataset used to build the random forest these are not
#' bag estimates.  The predicted classes that are output are directly
#' thus they are predictions resulting from the out of bag esimates.
#'
#' @param rf_obj a randomForest object of type regression
#' @param newdata data frame used to make predictions. Can be same as
#'                data.
#' @param breaks a numeric vector of values used to turn predictions
#' @param labels a character vector of labels for the classes
#' @param ordered logical indicating if categories are ordered (in or
#'                in labels)
#' @param type a character vector indicating whether total votes or p
#'              should be returned
#' @return returns a dataframe of probabilites or number of votes for
#' @export
class_prob_rf <- function(rf_obj,newdata,breaks,labels,ordered=FALSE,
                          type=c("probs","votes")){
  if(rf_obj$type!="regression"){
    stop("rf_obj is not a regression randomForest object.")
  }
  type <- match.arg(type)
  preds <- predict(rf_obj,newdata=newdata,predict.all=TRUE)
  preds_df <- data.frame(preds$individual)
  class_prob <- apply(preds_df,2, function(x) cut(x,breaks,labels))
  row.names(class_prob)<-row.names(preds_df)
  if(type=="probs"){
    class_prob <- apply(class_prob,1, function(x)
      table(factor(x,levels=labels, ordered=ordered))/rf_obj$ntree)
```

https://github.com/USEPA/LakeTrophicModelling

- include
  - R/ custom functions
  - man/ manual (documentation) how to use functions
  - NAMESPACE feature of R packages exports functions in a package

37 lines (30 sloc) | 1.25 KB

```
% Generated by roxygen2 (4.1.1): do not edit by hand
% Please edit documentation in R/class_prob_rf.R
\name{class_prob_rf}
\alias{class_prob_rf}
\title{Get Class Prediction Probabilities}
\usage{
class_prob_rf(rf_obj, newdata, breaks, labels, ordered = FALSE,
  type = c("probs", "votes"))
}
\arguments{
\item{rf_obj}{a randomForest object of type regression}

\item{newdata}{data frame used to make predictions. Can be same as original
data.}

\item{breaks}{a numeric vector of values used to turn predictions into classes}

\item{labels}{a character vector of labels for the classes}

\item{ordered}{logical indicating if categories are ordered (in order specified
in labels)}

\item{type}{a character vector indicating whether total votes or probabilites
should be returned}
}
\value{
returns a dataframe of probabilites or number of votes for each class
}
\description{
This function takes a regression randomForest and returns class probabilities
for a provided classifcation.  These probabilities are based on a full, new datase
If using the dataset used to build the random forest these are not out of
bag estimates.  The predicted classes that are output are directly from the randor
thus they are predictions resulting from the out of bag esimates.
}

```

https://github.com/USEPA/LakeTrophicModelling

Best practices

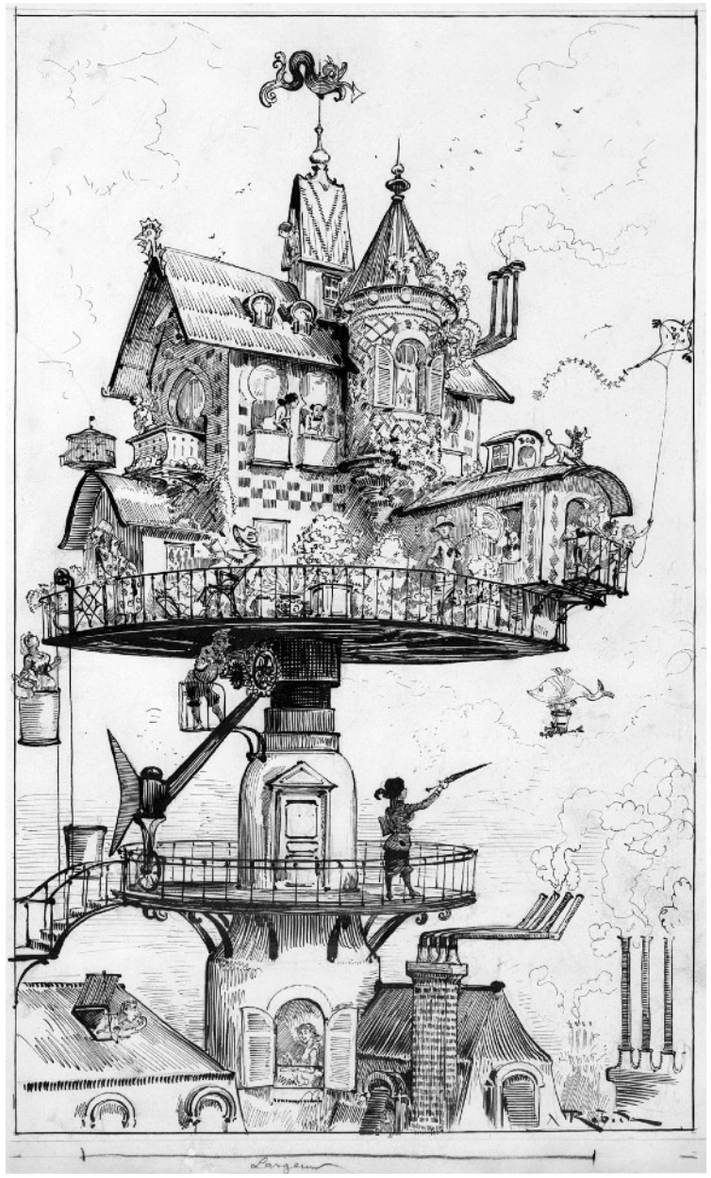
- Write programs for people, not computers
  - Write code that others can read and understand
    - Including your future self
- Code should not require to hold more than a handful of facts in memory
  - Functions !
- Make names consistent and meaningful
  - a, foo, results, results2…

Best practices

- Don’t repeat yourself
    - Modularize code rather than copy and paste
    - Re-use code instead of rewriting
        - Numerical integration
        - Matrix inversion

Best practices

- Document design and purpose, not mechanics
  - Interfaces and reasons and not implementations
  - If explanation is long and complicated
    - Restructure your code
- Embed the documentation for a piece of software in that software
  - Roxygen



Examples of real-world research compendia using R packages

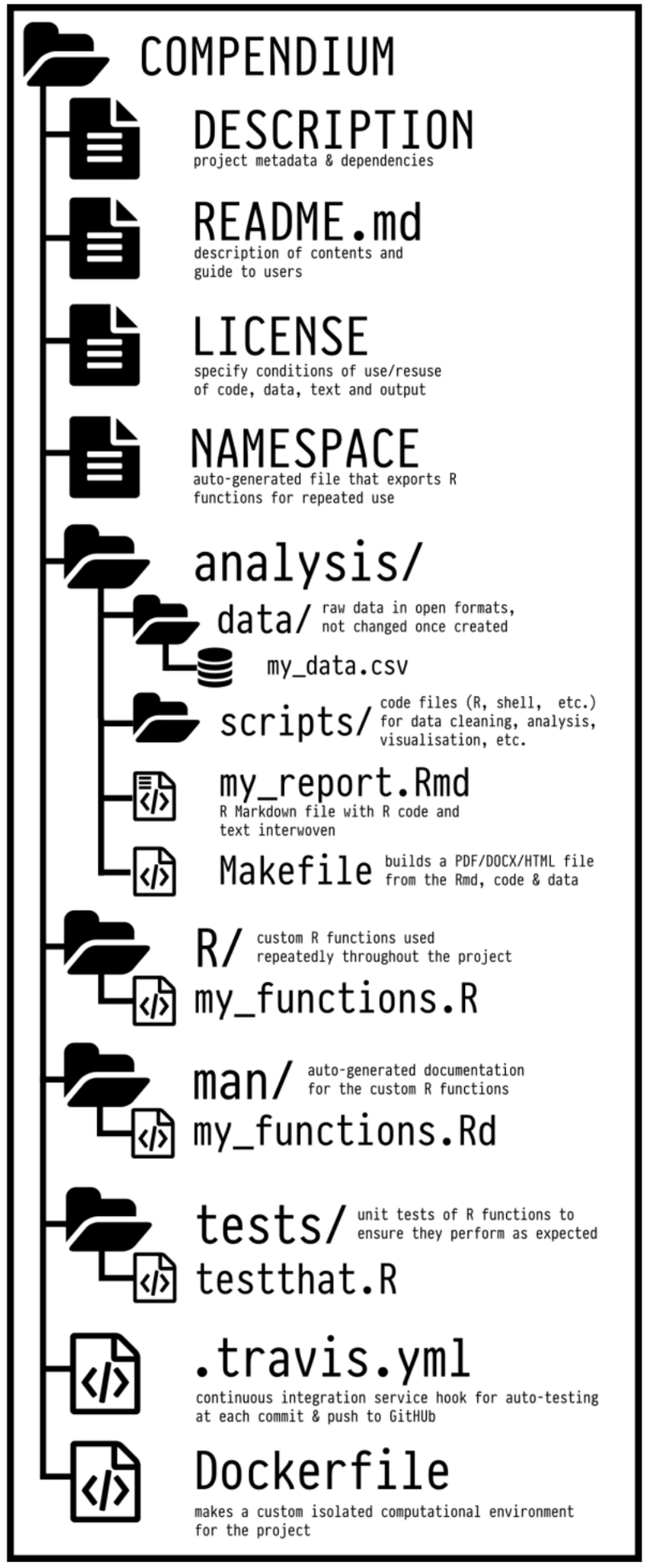
A LARGE compendium

Use ideas from software engineering

Generic Principles

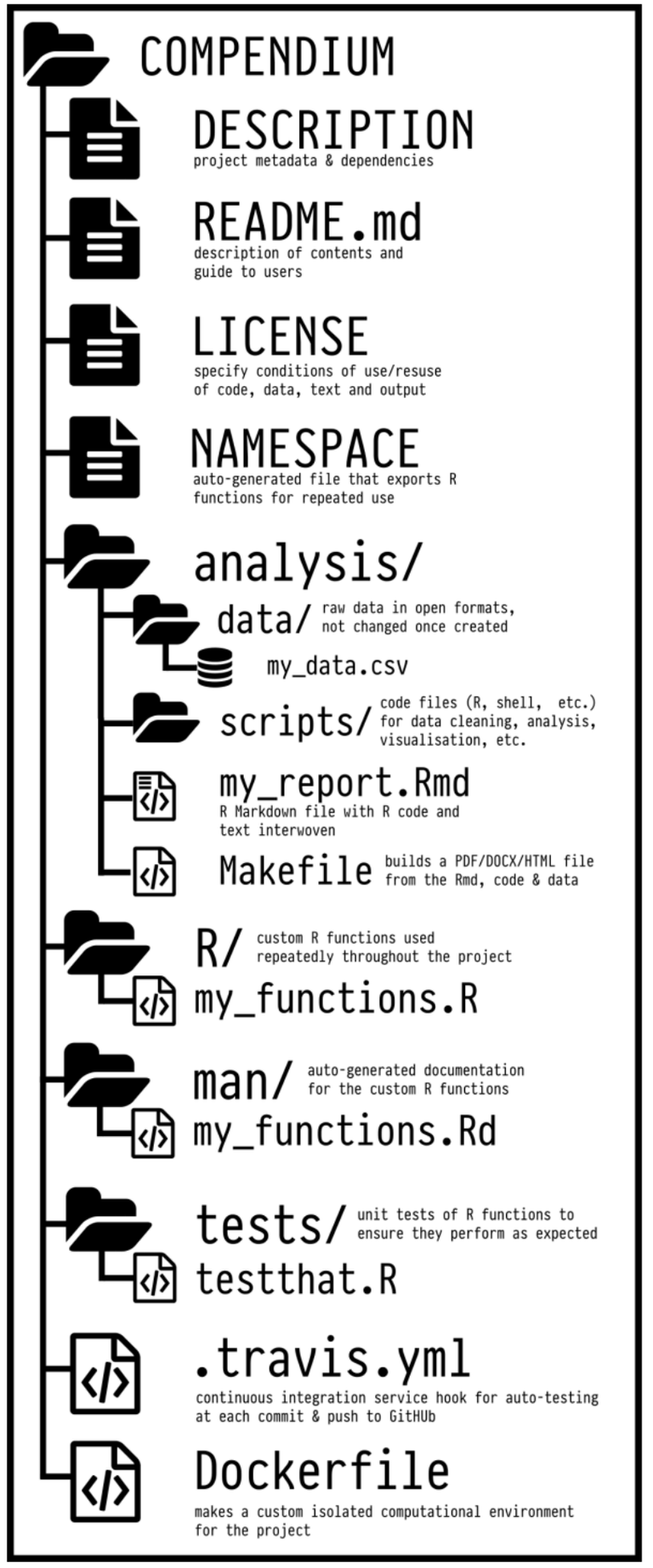
- Specify computational environment used for original analysis
    - e.g. document versions of libraries and packages

- includes tools from Software engineering
    - continuous integration via travis.yml



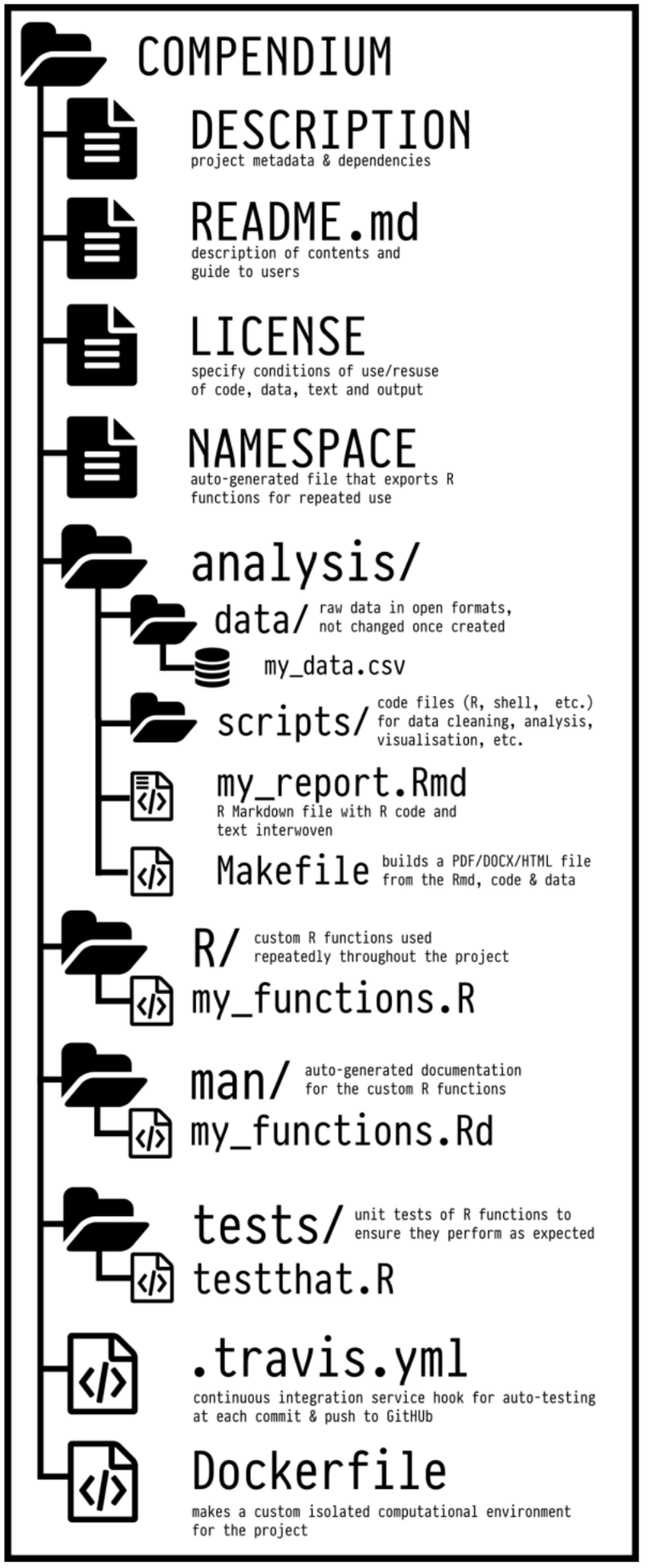
- Dockerfile

- Instructions to create a virtual environment for running the code in an isolated and portable context

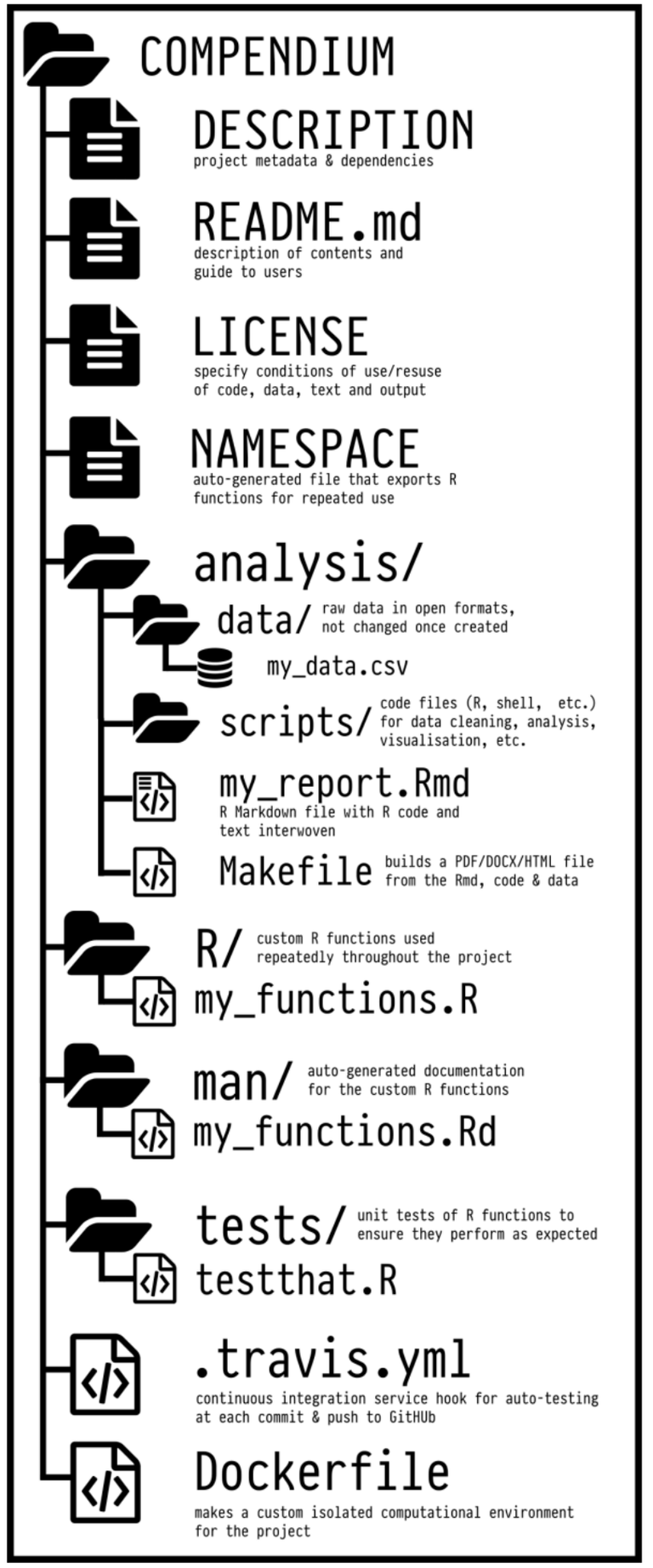


- Makefile

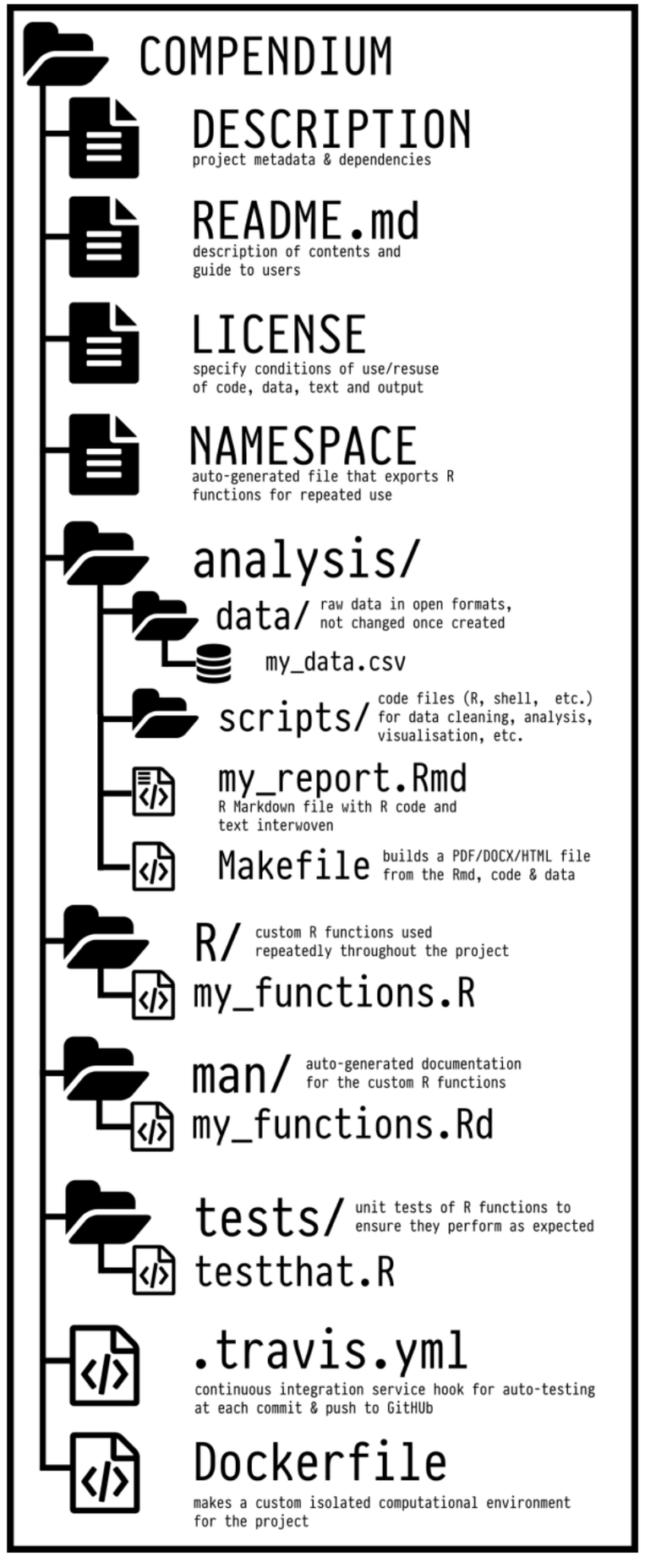
- instructions for building (executing) R code in the compendium avoiding unnecessary repetition



- tests/ Unit tests
    - code to test that specific function do what they are expected to do



- those tools solve problems of specifying computational environment and relationships between data, code, libraries and output



cboettig try again — Latest commit 79f712d on 5 Oct 2017

R	Include explicit returns to simplify conditional logic	2 years ago
inst/examples	avoid unnamed chunk	4 years ago
man	load .Rmd files with system.file from inst/examples instead	4 years ago
manuscripts	add codetools	6 months ago
tests	avoid RCurl test error	4 years ago
.Rbuildignore	update data file names to match figures	4 years ago
.drone.yml	supplement must be built in same command after manuscript.	4 years ago
.gitignore	dot files	4 years ago
.travis.yml	try again	6 months ago
.zenodo.json	add keywords	3 years ago
DESCRIPTION	add codetools	6 months ago
LICENSE	explicit license file	5 years ago
NAMESPACE	docs updated	4 years ago
README.md	Update README.md	6 months ago

README.md

nonparametric-bayes

DOI 10.5281/zenodo.13794

https:// github.com/cboettig/nonparametric-bayes

Best practices

- Plan for mistakes
    - Add assertions to programs to check their operation
- Automated testing
    - Compare against simplified cases, experimental data or results of earlier programs you trust
- http://r-pkgs.had.co.nz/tests.html

Best practices

- Optimize software only after it works correctly
- Write code in highest level language possible
- Use a profiler to identify bottlenecks

Best practices

- Collaborate
  - Use pre-merge code reviews
  - Use pair programming when bringing someone new up to speed
  - Use issue tracking tool (see GitHub)

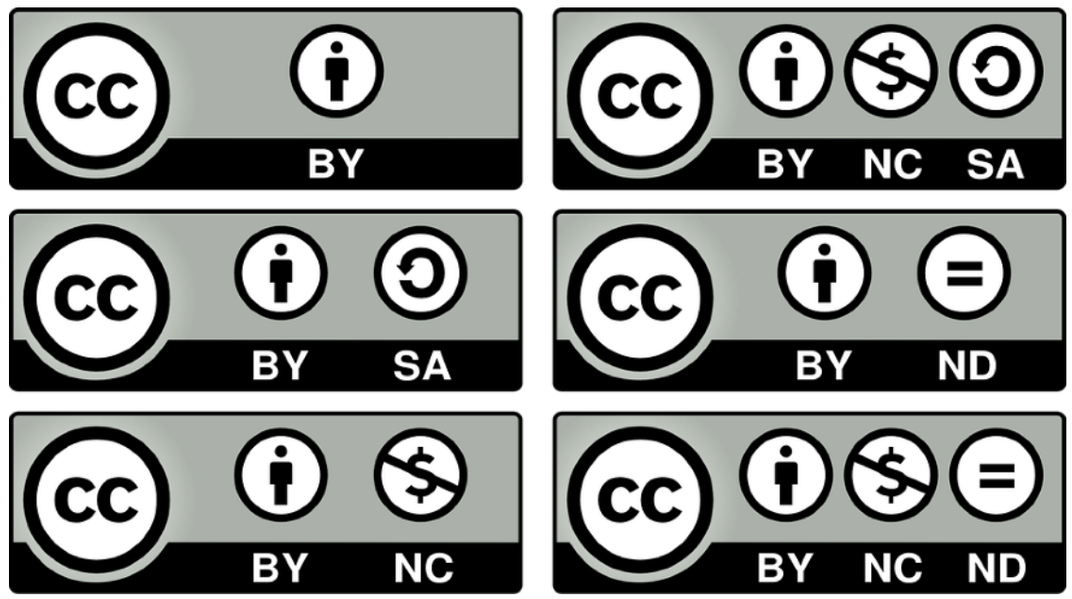
Sharing and Publishing

consider:

- Licensing
- Version control
- Persistence
- Metadata

License

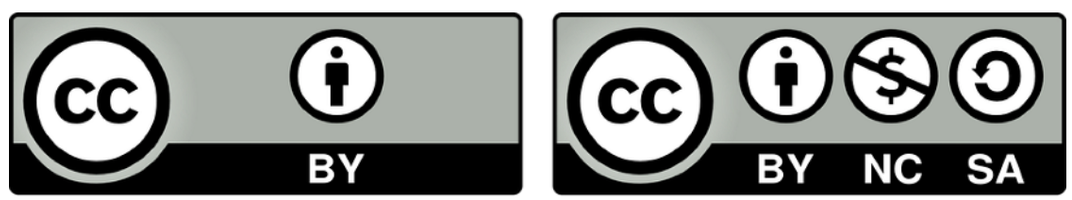
- Depends on your organization/ publisher
- Open software licenses
  - http://opensource.org
  - designed for software, and doesnt always work for other artefacts
    - many jurisdictions consider data as facts rather than creative work so they are not protected by IP copyright law

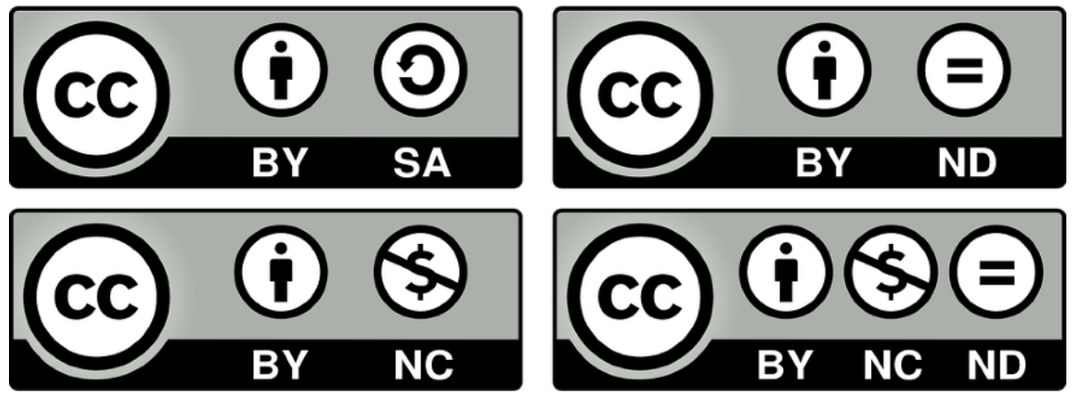


License

- data: CC-0

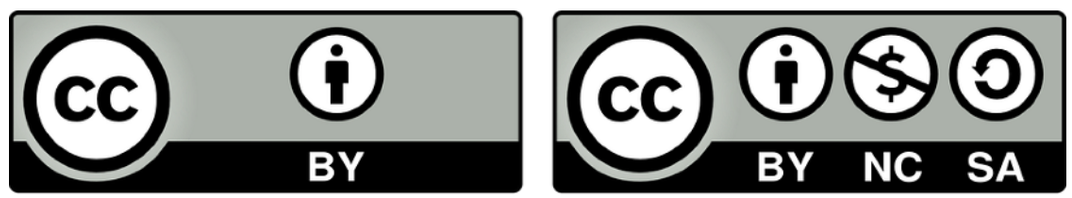
- Creative Commons Public Domain



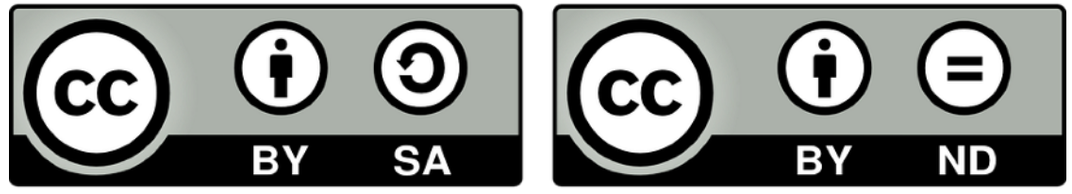


License

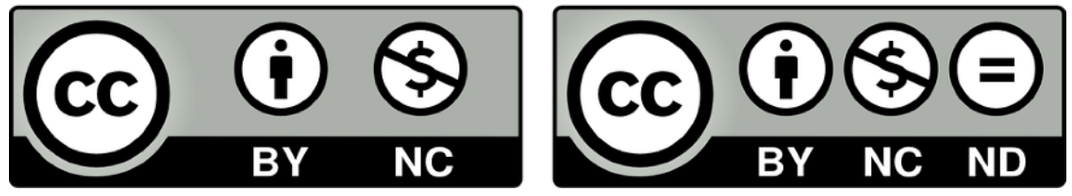
- documentation/manuscripts: CC-BY



- bioRxiv



- rXiv

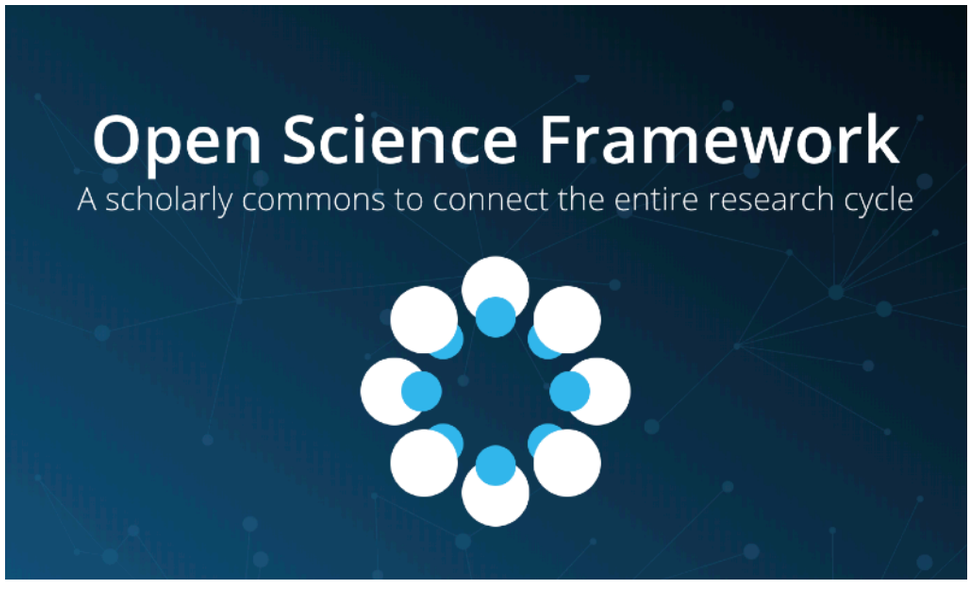


Version control

- Git is a good candidate
  - preserves history of changes
  - facilitates (private) collaborations, distribution and maintenance

Persistence

- use a repository with persistent URLs such as
  - DOI
  - zenodo.org
  - osf.io
  - Persistent and citable



Persistence

- you can continue to work on your stuff and the persistent link to a specific commit will make sure, that everybody reading your work can reproduce the exact same results

Git / Github

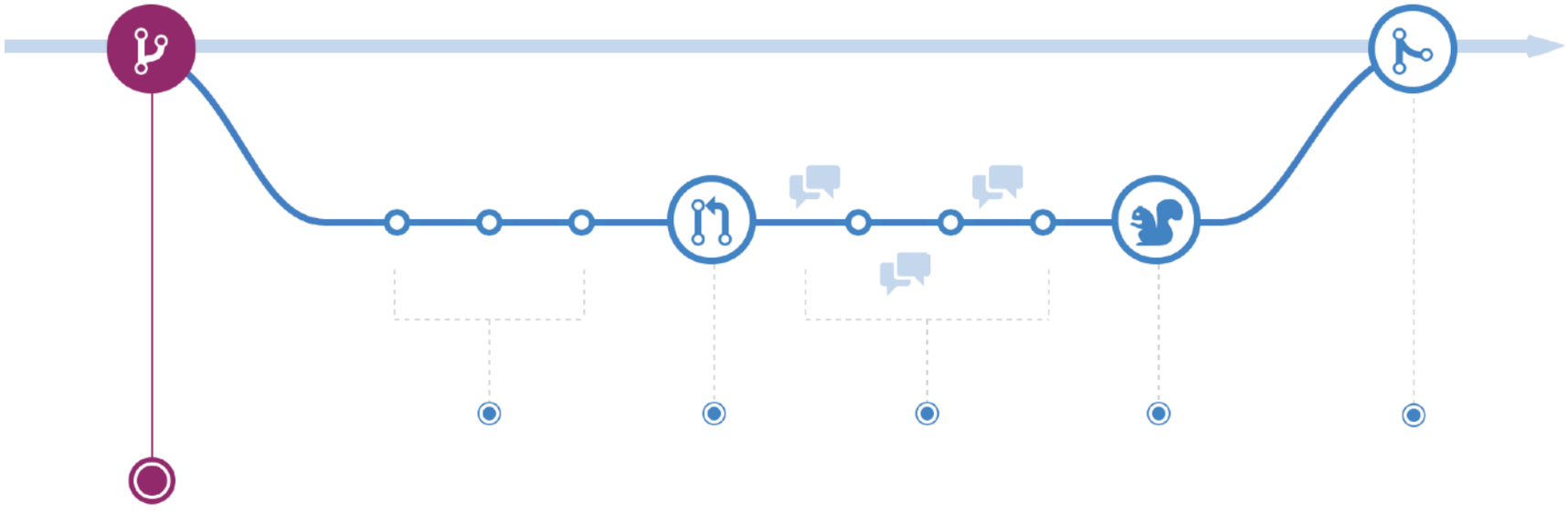
- Web-based hosting service for version controlled (git) projects

- Simple project management tools

Understanding the GitHub Flow

5 minute read Download PDF version

GitHub Flow is a lightweight, branch-based workflow that supports teams and projects where deployments are made regularly. This guide explains how and why GitHub Flow works.



https://guides.github.com/introduction/flow/

Docker

- specify environment with all dependencies necessary for analysis to run

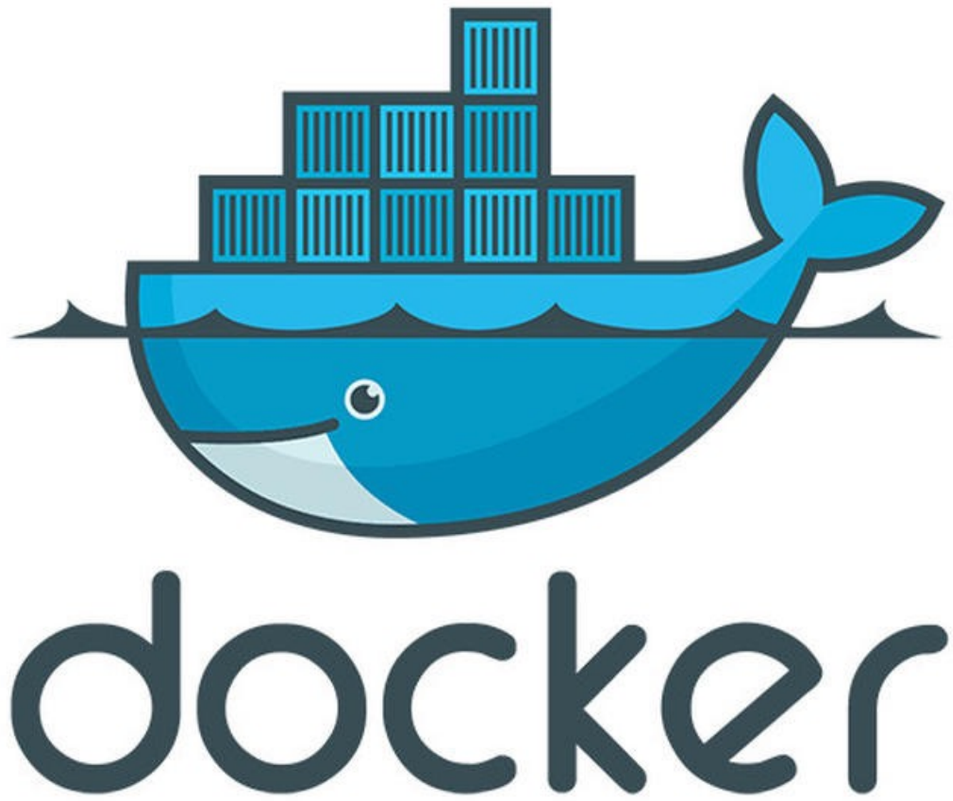
- lightweight virtualization

  - preserves package versions used for analysis

  - prevent breaking changes to influence your analysis

  - easy to share

- security?



Docker - alternatives

- packrat downloads source code of packages you have used

- devtools packages from Wickham and Chang 2016

```
> install.packages("reshape2")
```

If you completed the previous steps correctly, you just private package library. Let's take a snapshot to save th

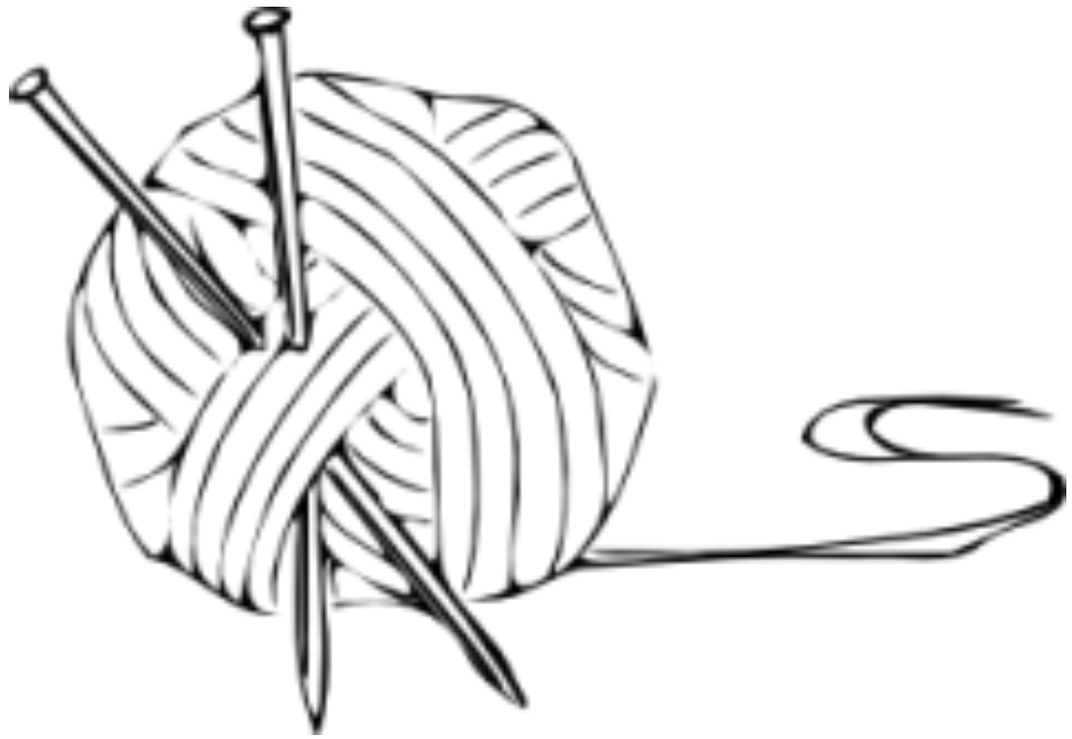
```
> packrat::snapshot()

Adding these packages to packrat:

             _
    plyr       1.8.1
    Rcpp       0.11.2
    reshape2   1.4
    stringr    0.6.2

Fetching sources for plyr (1.8.1) ... OK (CRAN cu
Fetching sources for Rcpp (0.11.2) ... OK (CRAN c
Fetching sources for reshape2 (1.4) ... OK (CRAN
Fetching sources for stringr (0.6.2) ... OK (CRAN
Snapshot written to '/Users/kevin/projects/babyna
```

Continuous integration

- each commit to Git repository triggers script to build your package (or manuscript)
- Travis, Drone, CircleCI
    - provide free remote CI services for public Github projects
    - provide badges for website that signal current state of compendium

Literate Programming

- Sweave (Leisch 2002)
- knitr and R markdown
- See also http://www.datacarpentry.org/



devtools

devtools: Tools to Make Developing R Packages Easier

Collection of package development tools.

Version:	1.13.5
Depends:	R (≥ 3.0.2)
Imports:	httr (≥ 0.4), utils, tools, methods, memoise (≥ 1.0.0), whisker, digest, rstudioapi (≥ 0.2.0), jsonlite, st
Suggests:	curl (≥ 0.9), crayon, testthat (≥ 1.0.2), BiocInstaller, Rcpp (≥ 0.10.0), MASS, rmarkdown, knitr, hun
Published:	2018-02-18
Author:	Hadley Wickham [aut], Jim Hester [aut, cre], Winston Chang [aut], RStudio [cph], R Core team [ctb
Maintainer:	Jim Hester <james.hester at rstudio.com>
BugReports:	https://github.com/hadley/devtools/issues
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
URL:	https://github.com/hadley/devtools
NeedsCompilation:	yes
Materials:	README NEWS
CRAN checks:	devtools results

rrtools

rrtools: Tools for Writing Reproducible Research in R

The goal of rrtools is to provide instructions, templates, and functions for making a basic compendium suitable for doing reproducible research with R. This package documents the key steps and provides convenient functions for quickly creating a new research compendium. The approach is based generally on Kitzes et al. (2017), and more specifically on Marwick (2017), Marwick et al. (2017), and Wickham's (2017) work using the R package structure as the basis for a research compendium.

- extends devtools for making basic compendium packages

Science writing

- rticles
  - https://github.com/rstudio/rticles
- citr
  - https://github.com/crsh/citr
- Bookdown
  - https://bookdown.org/

NATURE

A WEEKLY ILLUSTRATED JOURNAL OF SCIENCE

"To the solid ground
Of Nature trusts the mind which builds for aye."—WORDSWORTH

THURSDAY, NOVEMBER 4, 1869

NATURE: APHORISMS BY GOETHE

NATURE! We are surrounded and embraced by her: powerless to separate ourselves from her, and powerless to penetrate beyond her.

Without asking, or warning, she snatches us up into her circling dance, and whirls us on until we are tired, and drop from her arms.

She is ever shaping new forms: what is, has never yet been; what has been, comes not again. Everything is new, and yet nought but the old.

We live in her midst and know her not. She is incessantly speaking to us, but betrays not her secret. We constantly act upon her, and yet have no power over her.

The one thing she seems to aim at is Individuality; yet she cares nothing for individuals. She is always building up and destroying; but her workshop is inaccessible.

Her life is in her children; but where is the mother? She is the only artist; working-up the most uniform material into utter opposites; arriving, without a trace of effort, at perfection, at the most exact precision, though always veiled under a certain softness.

Each of her works has an essence of its own; each of her phenomena a special characterisation: and yet their diversity is in unity.

She performs a play; we know not whether she sees it herself, and yet she acts for us, the lookers-on.

Incessant life, development, and movement are in her, but she advances not. She changes for ever and ever, and rests not a moment. Quietude is inconceivable to her, and she has laid her curse upon rest. She is firm. Her steps are measured, her exceptions rare, her laws unchangeable.

She has always thought and always thinks; though not as a man, but as Nature. She broods over an all-comprehending idea, which no searching can find out.

Mankind dwell in her and she in them. With all men she plays a game for love, and rejoices the more they win. With many, her moves are so hidden, that the game is over before they know it.

That which is most unnatural is still Nature; the stupidest philistinism has a touch of her genius. Whoso cannot see her everywhere, sees her nowhere rightly.

She loves herself, and her innumerable eyes and affections are fixed upon herself. She has divided herself that she may be her own delight. She causes an endless succession of new capacities for enjoyment to spring up, that her insatiable sympathy may be assuaged.

She rejoices in illusion. Whoso destroys it in himself and others, him she punishes with the sternest tyranny. Whoso follows her in faith, him she takes as a child to her bosom.

Her children are numberless. To none is she altogether miserly; but she has her favourites, on whom she squanders much, and for whom she makes great sacrifices. Over greatness she spreads her shield.

She tosses her creatures out of nothingness, and tells them not whence they came, nor whither they go. It is their business to run, she knows the road.

Her mechanism has few springs—but they never wear out, are always active and manifold.

The spectacle of Nature is always new, for she is always renewing the spectators. Life is her most exquisite invention; and death is her expert contrivance to get plenty of life.

She wraps man in darkness, and makes him for ever long for light. She creates him dependent upon the earth, dull and heavy; and yet is always shaking him until he attempts to soar above it.

B

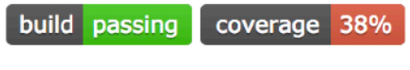
Templates

Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

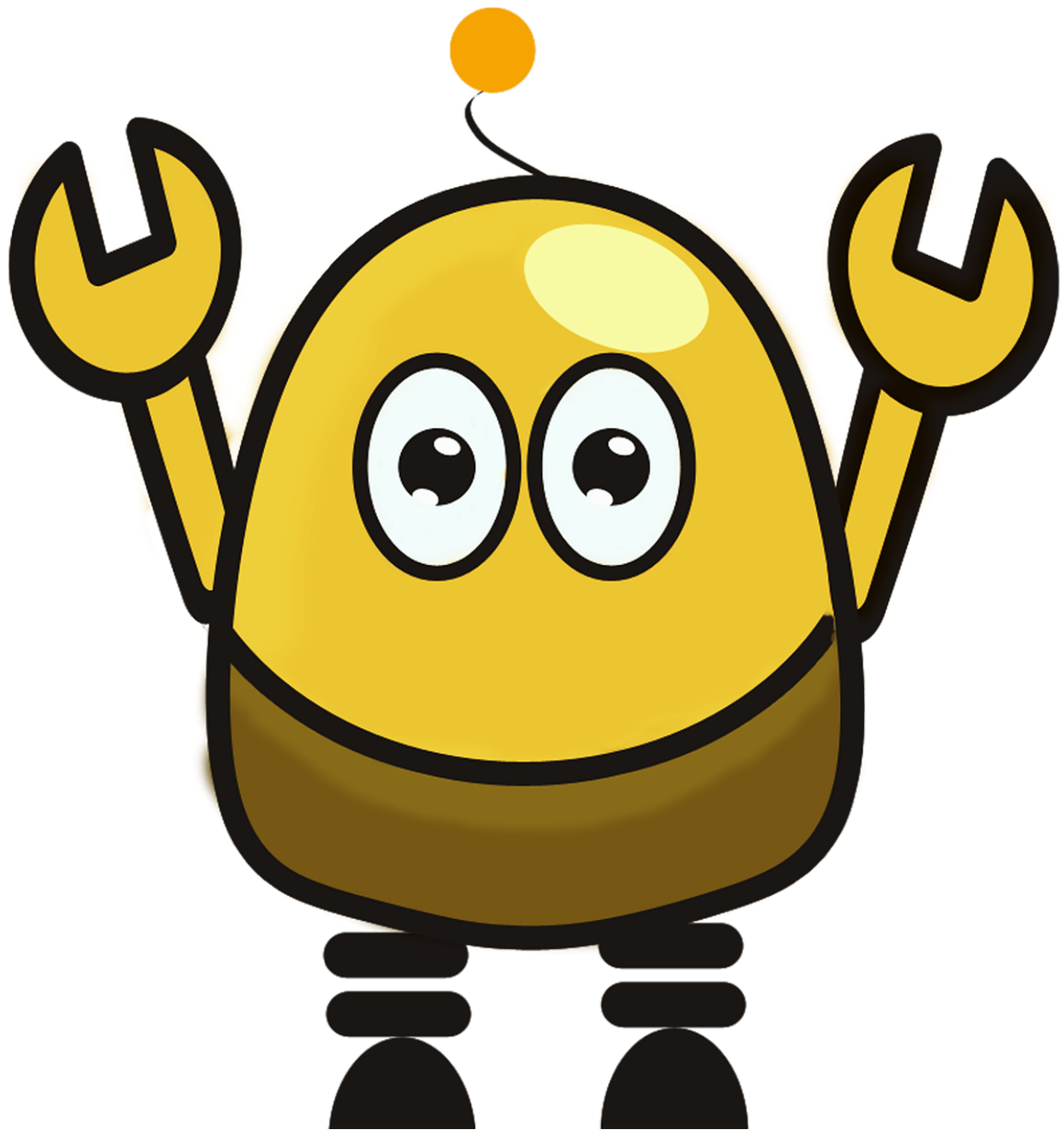
cboettig draft for `template` function to deploy template. ··· Latest commit **6432b97** on 6 Nov 2017

R	draft for `template` function to deploy template.	5 months ago
man	draft for `template` function to deploy template.	5 months ago
tests	add docs and a dummy test	3 years ago
vignettes	add manuscript-style vignette	3 years ago
.Rbuildignore	add manuscript-style vignette	3 years ago
.gitignore	add manuscript-style vignette	3 years ago
.travis.yml	include ORCID	5 months ago
CONTRIBUTING.md	add the previous contributing guidelines	3 years ago
DESCRIPTION	draft for `template` function to deploy template.	5 months ago
LICENSE	minor formatting tweaks	3 years ago
NAMESPACE	draft for `template` function to deploy template.	5 months ago
README.Rmd	add coveralls badge	3 years ago
README.md	add coveralls badge	3 years ago
cran-comments.md	revised template, using new devtools functions	3 years ago
template.Rproj	fix Authors@R line	3 years ago

README.md



https://github.com/cboettig/template



Conclusions

- General principles of research compendia
  - Adhere (and define) to a standard
  - Clear separation of data, method, and output
  - Specify computational environment
  - Document
  - Use version control

Keep your future self happy !

- for someone working primarily in R, a compendium based on R package standards is natural and efficient

- if you work in other languages, you can draw ideas from the general principles presented here

Treat your code and data as much as a product of your research as articles.

References

- Marwick, Boettiger and Mullen "Packaging data analytical work reproducibly using R (and friends)" 2018 [https://peerj.com/preprints/3192.pdf]

- Wilson, Greg, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, and Tracy K. Teal. 2016. “Good Enough Practices in Scientific Computing.” arXiv [cs.SE]. arXiv. http://arxiv.org/abs/1609.00037.

- Wilson, Greg, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, et al. 2014. “Best Practices for Scientific Computing.” PLoS Biology 12 (1): e1001745.

- https://github.com/benmarwick/rrtools
- https://github.com/cboettig/template