From scripts to packages: how to be disciplined about your R code

Theodor Adrian Balan

Leiden University Medical Center, The Netherlands

- Leic


Leeuwarden
Groningen
E22
E232
Alkmaar
E22
Amsterdam
E232
The Hague
Netherlands
Arnhem
Rotterdam
E31
E311
E25
Eindhoven
Essen
Bruges
Antwerp
Düsseldor
E40
Ghent
E19
3
E25
61

Dutch climate

Spring

Summer

Autumn

Winter

```
Theodors-MacBook-Pro:Ascertainment theodor$ ls *.R
EB_frailties.R
EB_frailty_estimates.R
agsp_asc.R
agsp_noasc.R
correlation_estimates.R
crazyestimation.R
cumiudexes.R
data_analysis_final.R
expEstTest.R
frailasc.R
frailasc_iterative-Teddy's MacBook Pro.R
frailasc_iterative.R
frailasc_iterative_2proc.R
frailasc_iterative_asc.R
frailasc_iterative_asc_2proc.R
frailasc_iterative_asc_final.R
frailasc_iterative_current.R
frailasc_iterative_for2proc.R
frailasc_iterative_semipar.R
frailasc_iterative_tdep.R
frailasc_next.R
frailasc_nofrail.R
frailasc_nofrail_iterative.R
frailasc_th0.R
frailsp.R
frailsp_asc.R
frailsp_asc_corrected.R
frailsp_asc_wt.R
frailsp_wt.R
frailtyEB.R
frailty_pwc_direct_asc_beta.R
frailty_pwc_timedep.R
gendat_one_cov.R
give_data_1cov_varbeta.R
give_data_2cov.R
give_data_2cov_ties.R
give_data_lambdar.R
give_data_lambdar_beta1.R
hub.shared.frailty.profile.theta.R
interactive_data_analysis.R
iterative_try.R
lexis.R
muie.R
new_playBHD.R
playwithBHD.R
plot_toyexample.R
problem_illustration.R
read_simulations.R
rev_bhplot_misspec.R
rev_check_coverage_theta.R
rev_error_check.R
rev_readsim.R
rev_readsim_f.R
rev_sharksim.R
rev_sharksim_05.R
rev_sharksim_1.R
rev_simulation.R
shared.frailty.parametric.R
shared.frailty.profile.theta.R
sim_bias_varybeta.R
sim_interval_asc.R
simth05_shark.R
simulation_example.R
simulation_largesample.R
simulation_organized.R
split_functions.R
supertry.R
trash_frailasc_function.R
trash_frailasc_function_v2.R
trash_frailasc_function_v3.R
trash_frailasc_function_vBeta.R
truncate_for_sim.R
validation_sim_oldvsnew.R
```

```
Theodors-MacBook-Pro:Ascertainment theodor$ ls *.r
brute.force.frailty.r           sftheta.r
frailty_pwc.r                   sfthetaasc.r
frailty_pwc_direct.r            shared.frailty.r
frailty_pwc_direct_asc.r        sim_bias.r
give.data.r                     try.ascertainment.r
give.data2.r                    working_f_frailtypwc.r
readdata.r
```

- Code is

- How to communicate with another **human** what you want the computer to do

- Purpose of R

- To spend the least amount of time possible in R.

- Programming mantra

- Constructive lazyness

- Arrange things so that you have to do a minimal amount of work in the long run.

Basic discipline

- Consistent code style
- (almost) everything should be in a function
- Short functions that do relatively simple things
- Functions called in a “main” R script that takes a journey from data to results when ran
- Comments and documentation
- Rstudio projects

Advanced discipline

- Never modify the original data
- Anything that *might* change in the future should be an argument of a **fun**ction
- Code should be expressive
- Proper data manipulation tools (dplyr & tidyr / data.table)
- Maintain an Rmarkdown file that has a working example of how to use the functions

Why discipline?

- Write as if someone **else** will have to read and understand the code

You now

You in 6 months

You now

Who is the crazy person that wrote this st?**

You in 6 months

You now

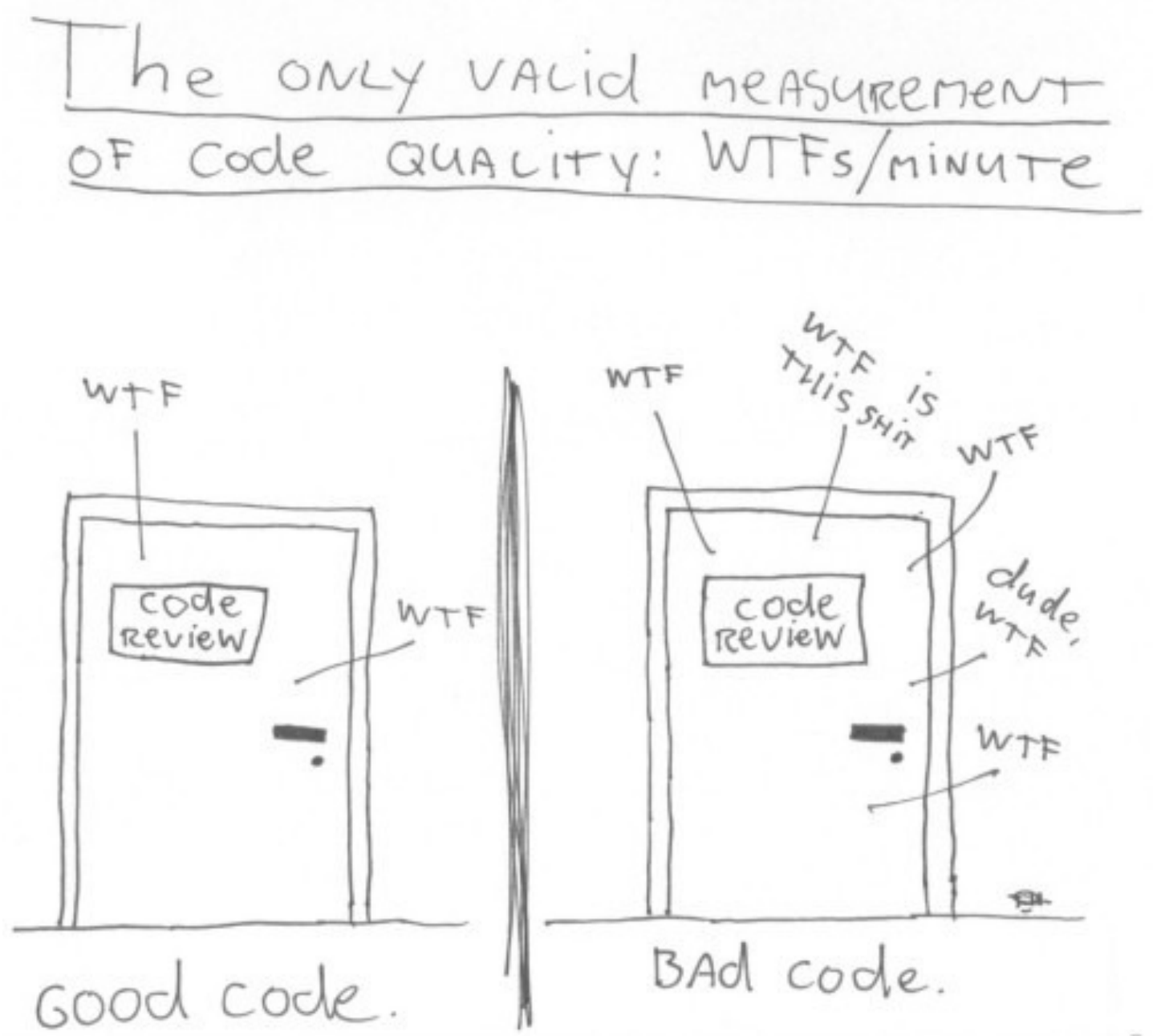
This will take me so much time!

You in 6 months

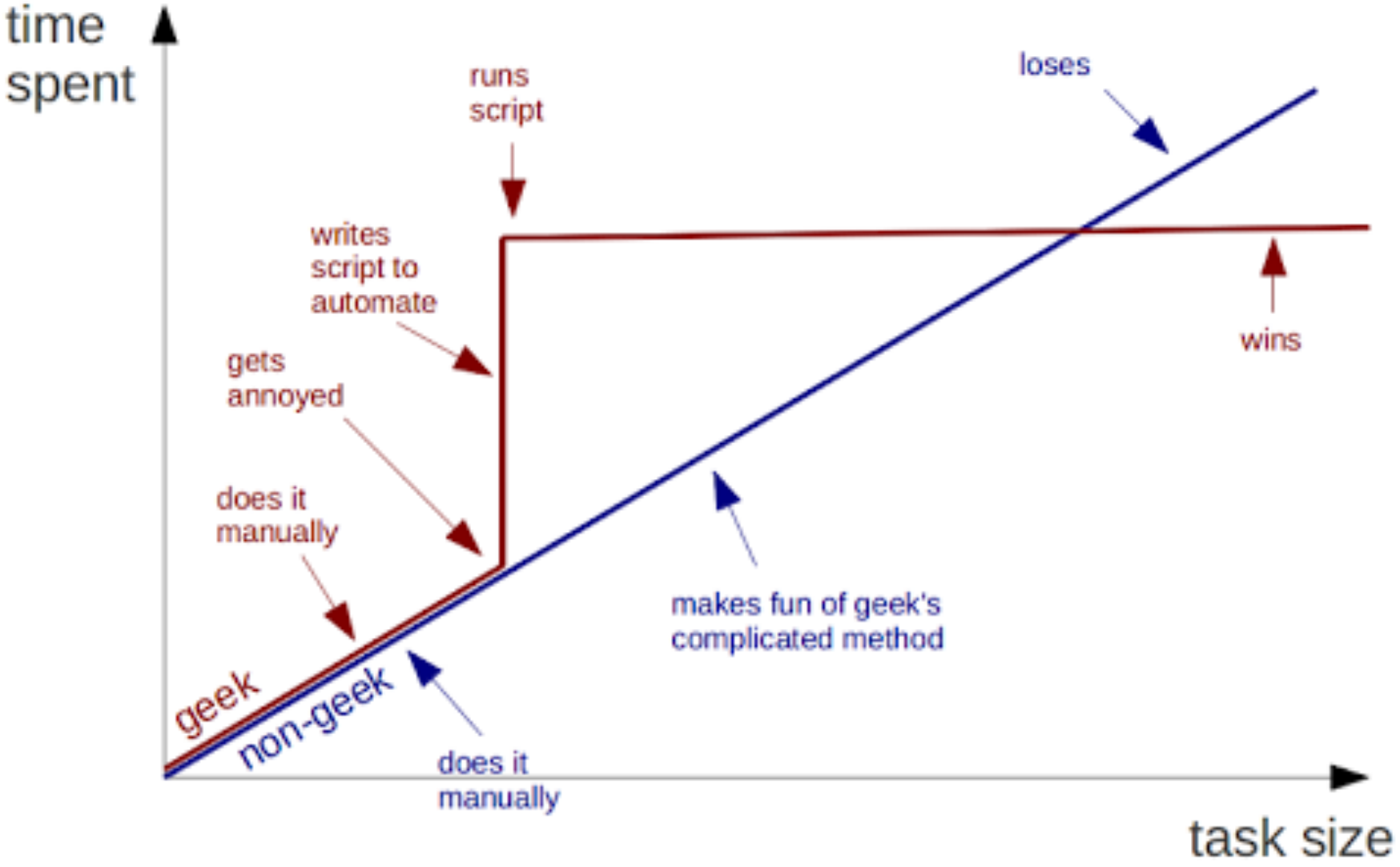
You now

Thanks past me for writing code that is easy to understand and saving my time!

You in 6 months



Geeks and repetitive tasks



Keep it manageable

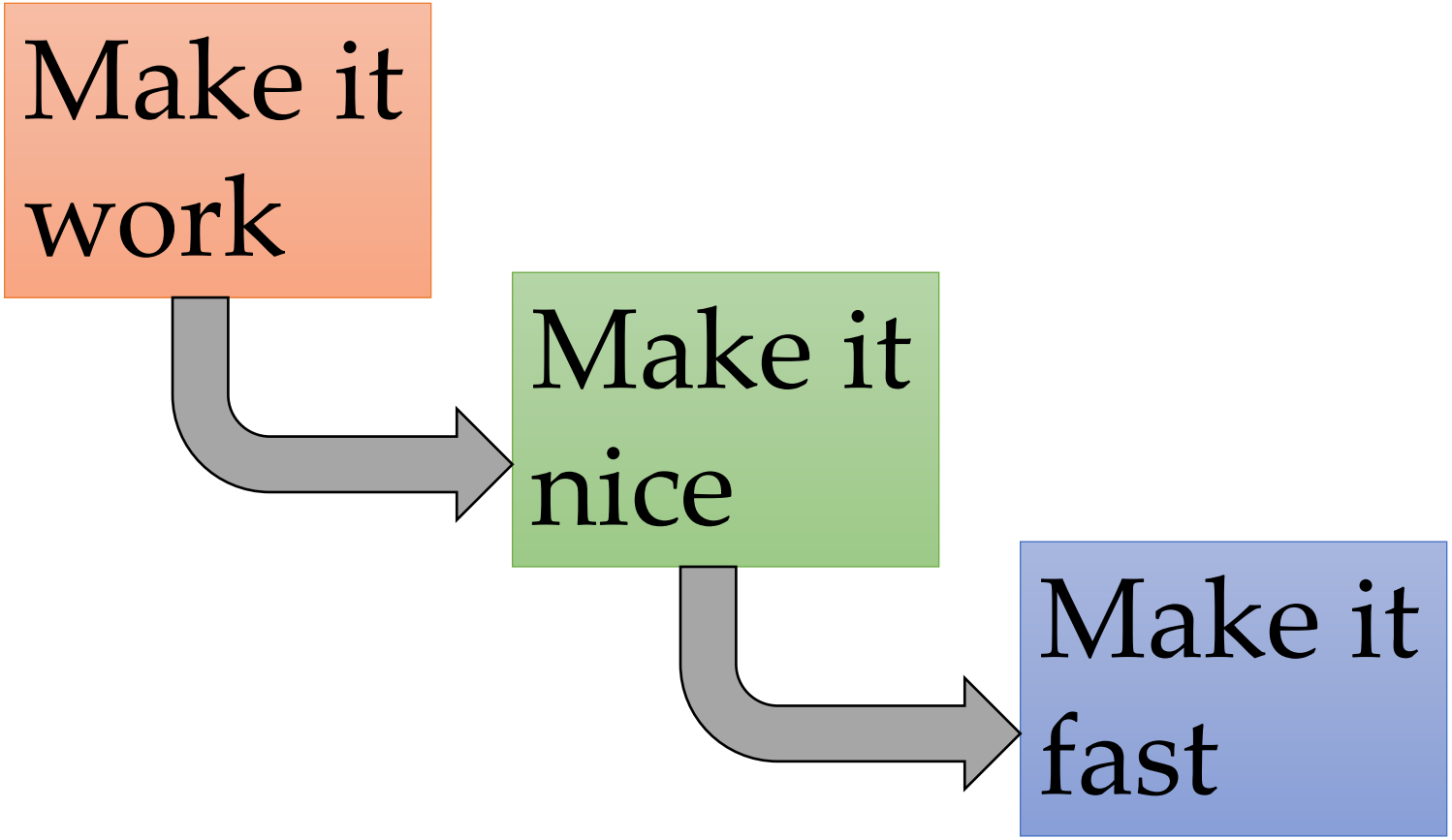
- Avoid redundant code (copy-paste)
- Re-use functions!

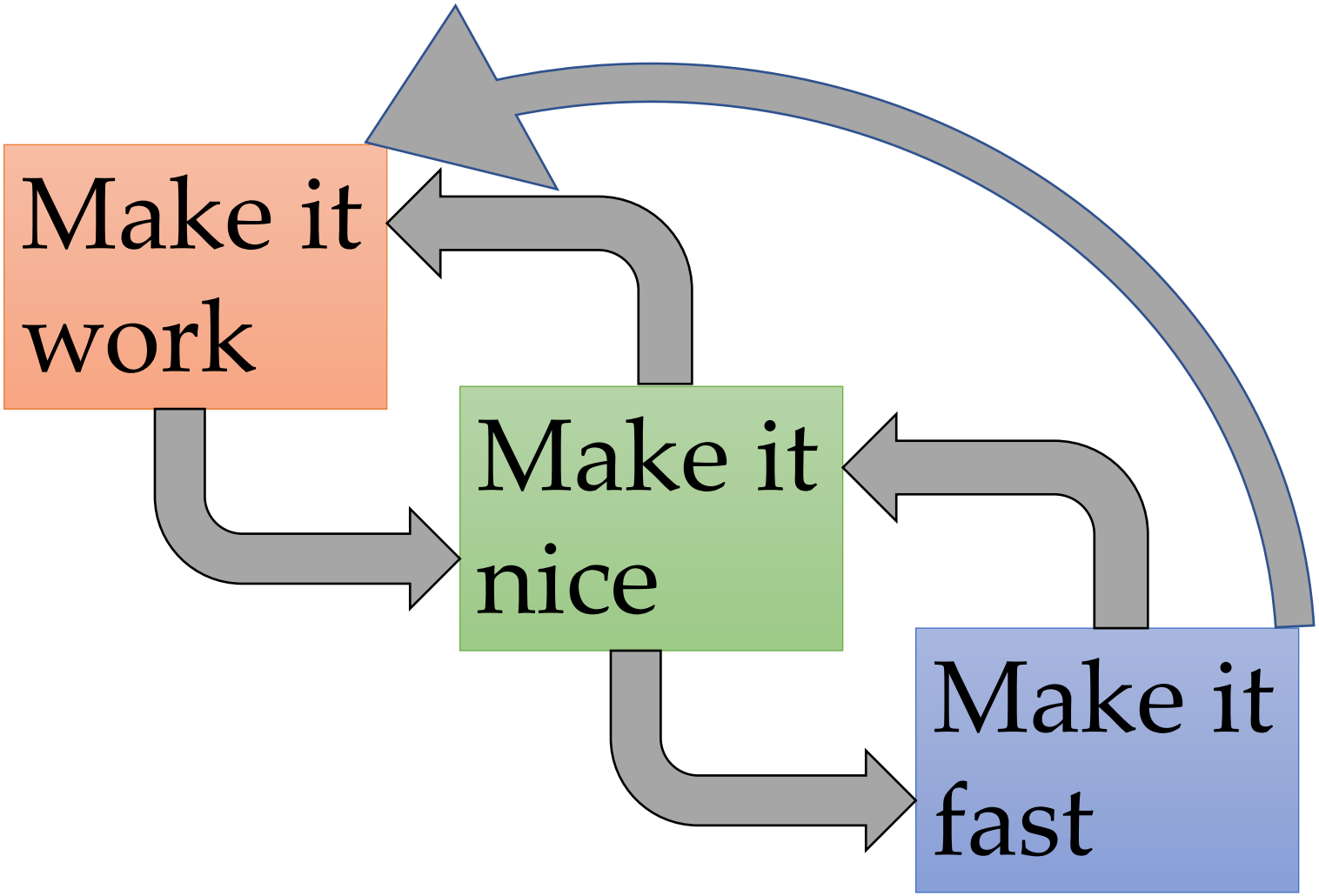
- **Git**(Hub) and version control:
  - Reduce the amount of files
  - Keep track of changes
  - Easy to publish code

Make it work

Make it nice

Make it fast


Make it work
Make it nice
Make it fast


Make it work
Make it nice
Make it fast

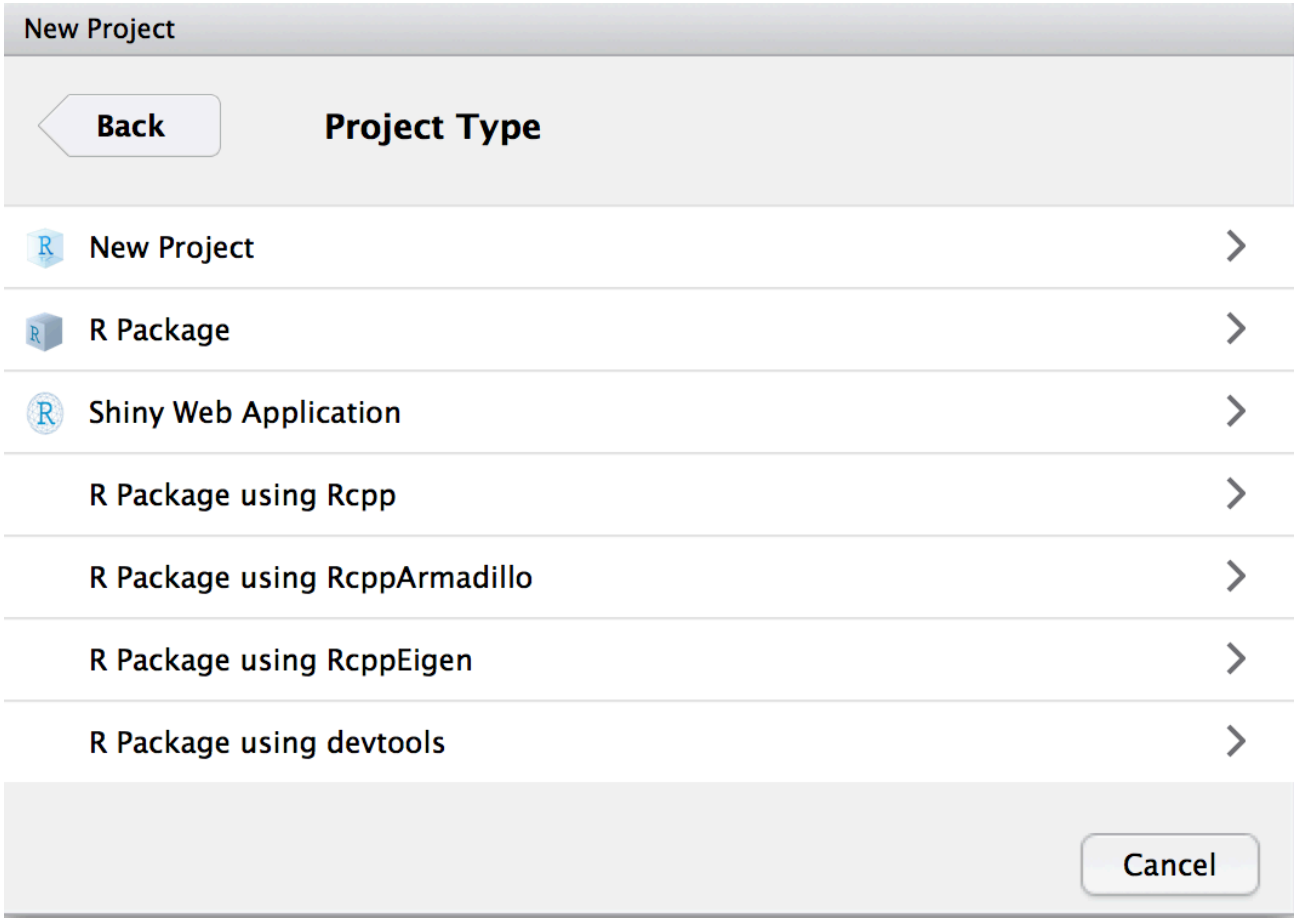
For larger projects, take time to think first

Write your whole program in words (pseudo-code) with pen and paper first

R packages

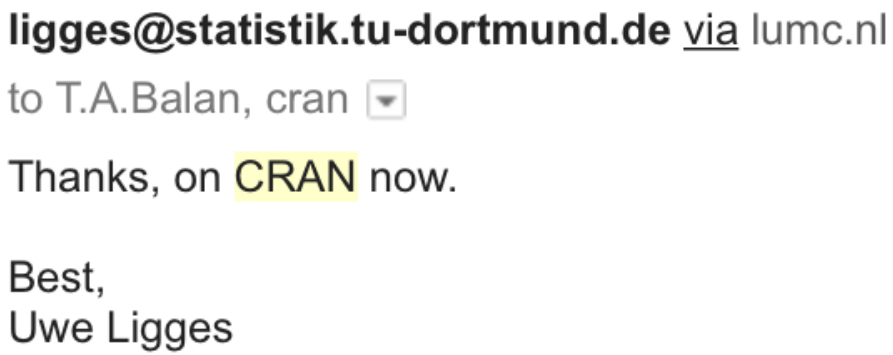
- It's very easy to make an R package in Rstudio (see also **devtools** package)
- Function documentation is easy with **roxygen2**
- Git features are built-in to Rstudio
- Automatic testing features incl. running all examples help with "bad" behaviour
- Longer package documentation (vignettes) with Rmarkdown

R packages


New Project
Back
Project Type
New Project
R Package
Shiny Web Application
R Package using Rcpp
R Package using RcppArmadillo
R Package using RcppEigen
R Package using devtools
Cancel

Sharing is caring

- Packages don't have to be complicated, they just have to be useful
- Anyone can install your package from Github (with **devtools**)
- Fixing warnings, proper documentation and examples, you can send it to CRAN




HACKERMAN™

Conclusion

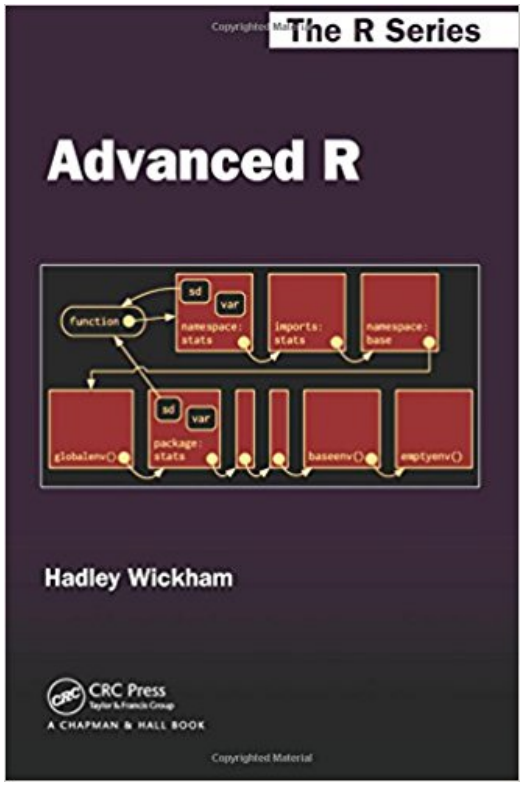
+It pays off in the long run to learn new tools

+Writing good code can help with having more time for doing statistics

+Think about data, models, methods etc.

+Good for visibility in science

Resources



stackoverflow

Conclusion

+It pays off in the long run to learn new tools

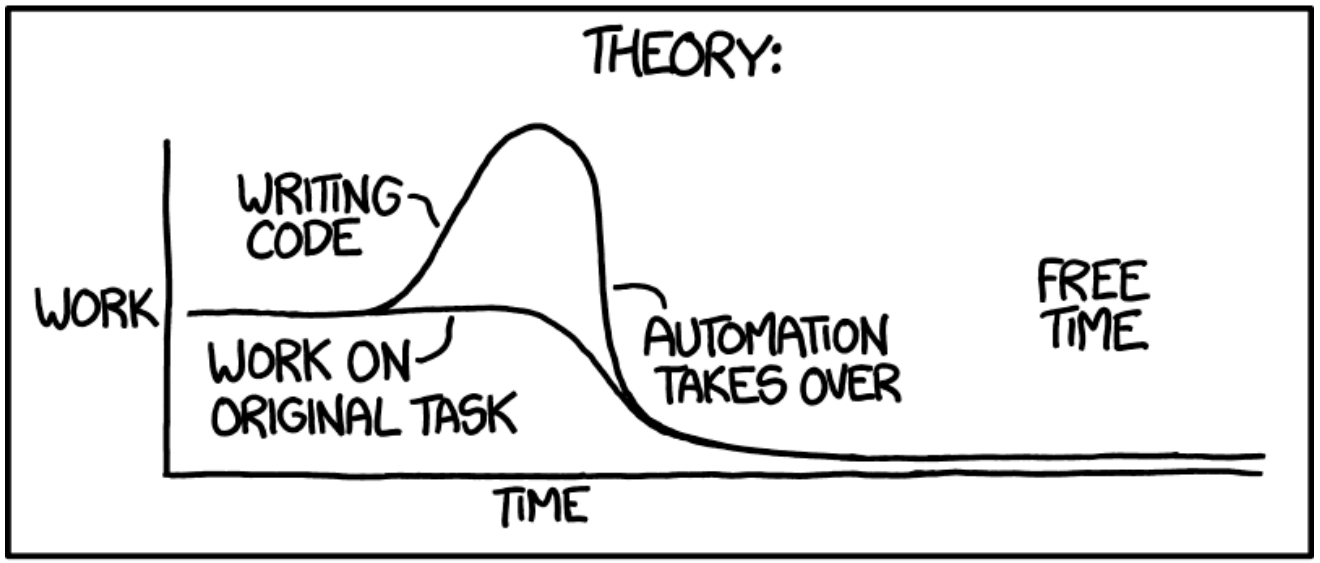
+Writing good code can help with having more time for doing statistics

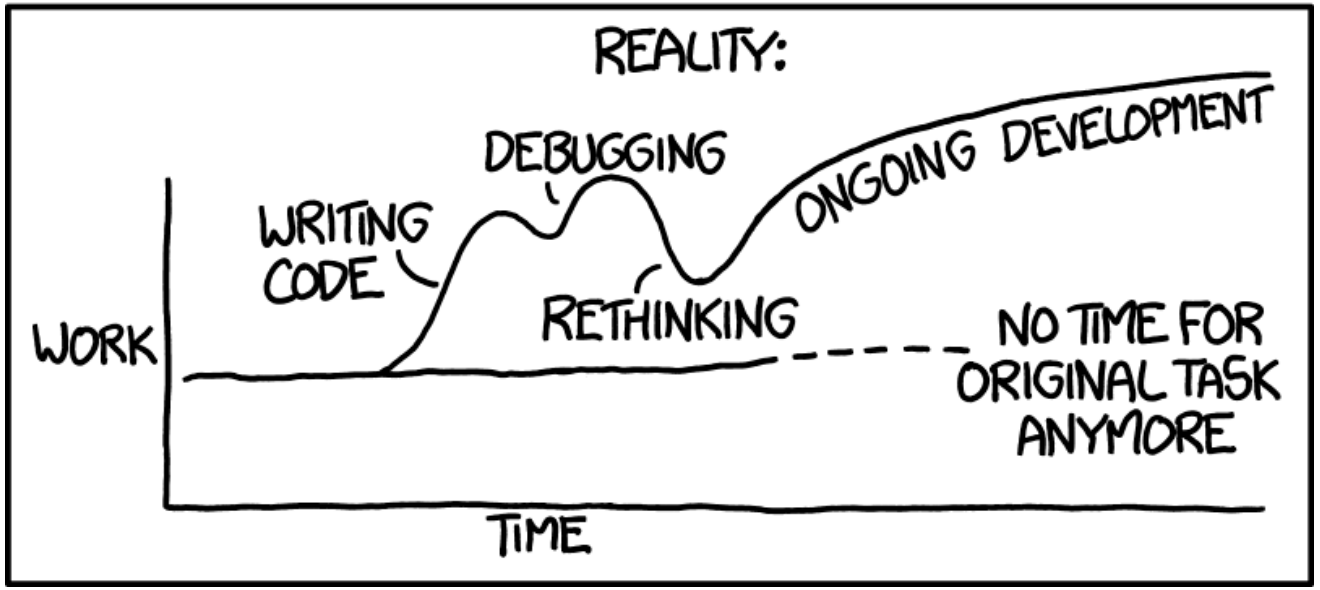
+Think about data, models, methods etc.

+Good for visibility in science

- You might start to really like it

"I SPEND A LOT OF TIME ON THIS TASK.
I SHOULD WRITE A PROGRAM AUTOMATING IT!"





Are you too busy to improve?

Håkan Forss @hakanforss http://hakanforss.wordpress.com

This illustration is inspired by and in part derived from the work by Scott Simmerman, "The Square Wheels Guy" http://www.performancemanagementcompany.com/