Coding practices

Hunyong Cho

Outline

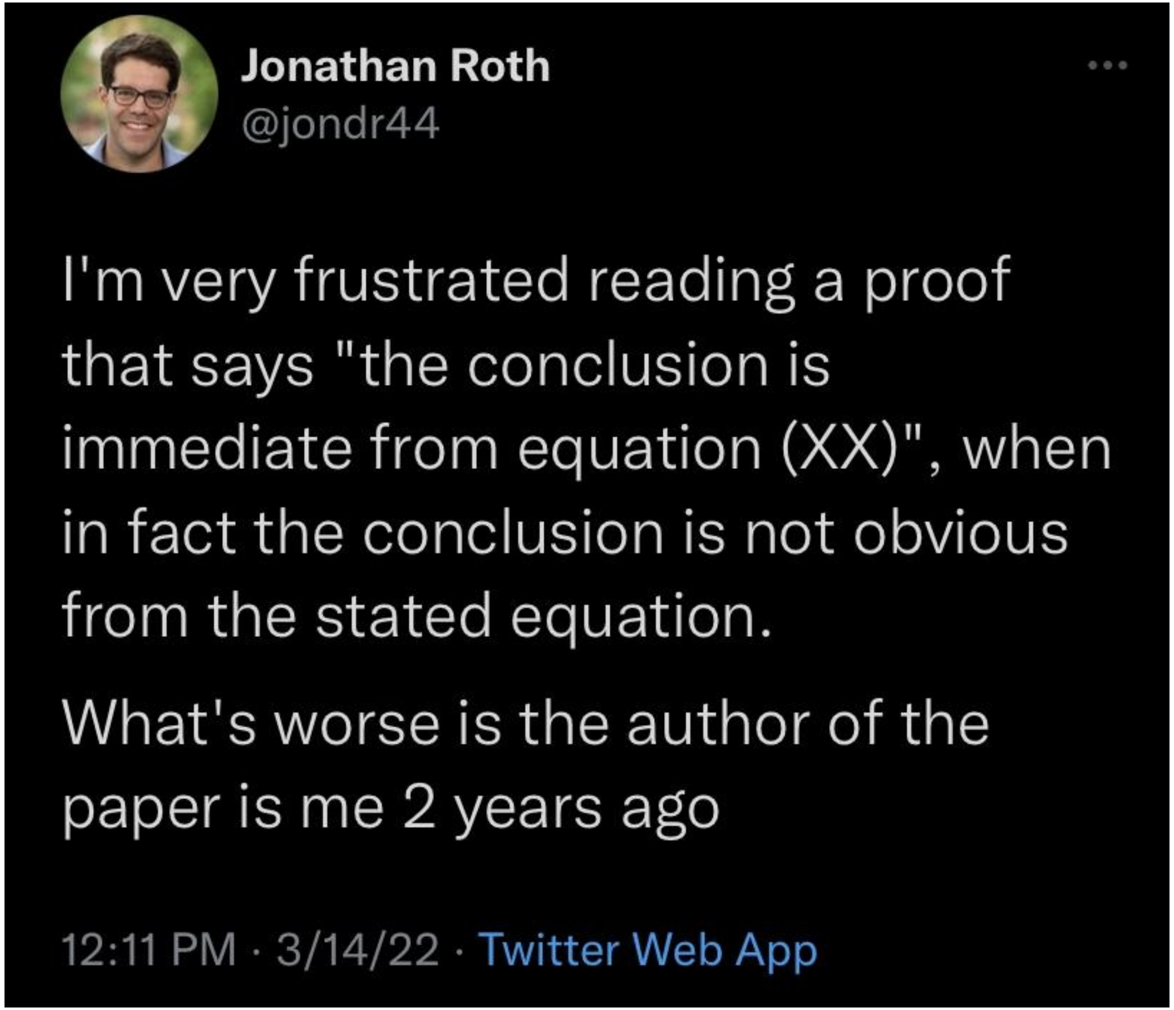
- Principles
- Examples that need improvement
- Tips for R programming

Code for yourself a month later

“Ugh.. The code is a complete mess.”

Sorry, but you wrote the code.

You are the author of your code, but are also your future reader.
Be kind to your future self.



Jonathan Roth
@jondr44

I'm very frustrated reading a proof that says "the conclusion is immediate from equation (XX)", when in fact the conclusion is not obvious from the stated equation.

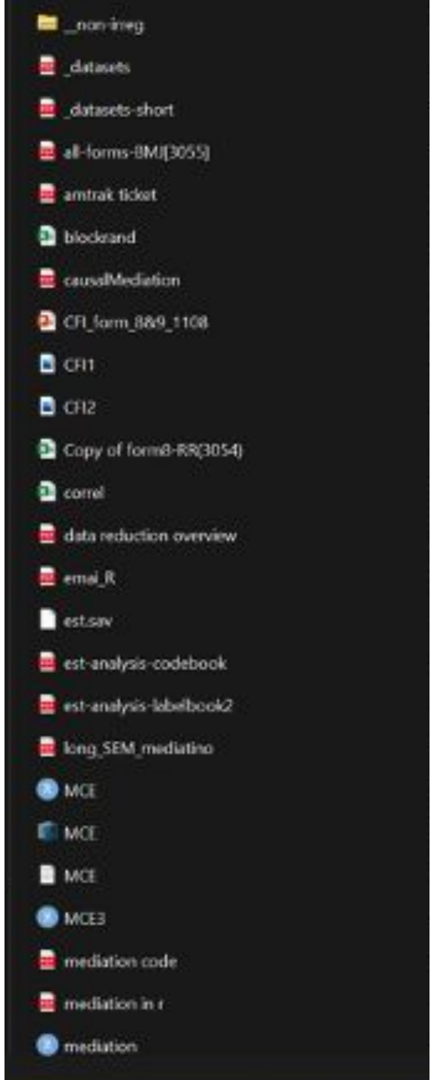
What's worse is the author of the paper is me 2 years ago

12:11 PM · 3/14/22 · Twitter Web App

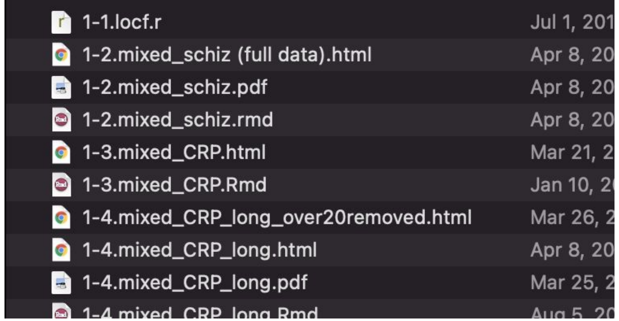
Some coding principles

● Reproducibility	No human interaction
● Validity	Unit validation (don’t write a lengthy code/function and hope it works at once!)
● Readability	KISS (keep it simple, stupid) Meaningful names, commenting, pipes, indentation
● Efficiency	Dry coding (Don’t repeat yourself), utilize functions Practice abstraction (but not too much!)

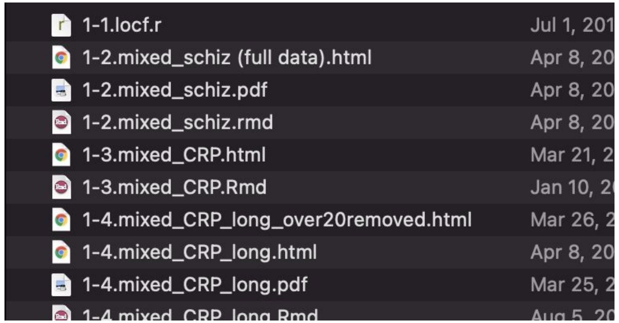
A coding project structure



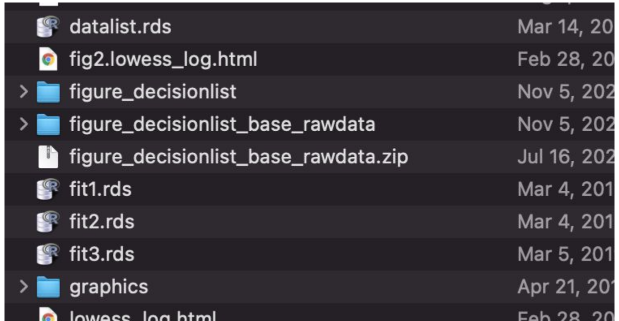
```
_non-imeg
_datasets
_datasets-short
all-forms-BMJ[3055]
amtrak ticket
blockrand
causalMediation
CFI_form_8&9_1108
CFI1
CFI2
Copy of form8-RR(3054)
correl
data reduction overview
emai_R
est.sav
est-analysis-codebook
est-analysis-labelbook2
long_SEM_mediatino
MCE
MCE
MCE
MCE3
mediation code
mediation in r
mediation
```



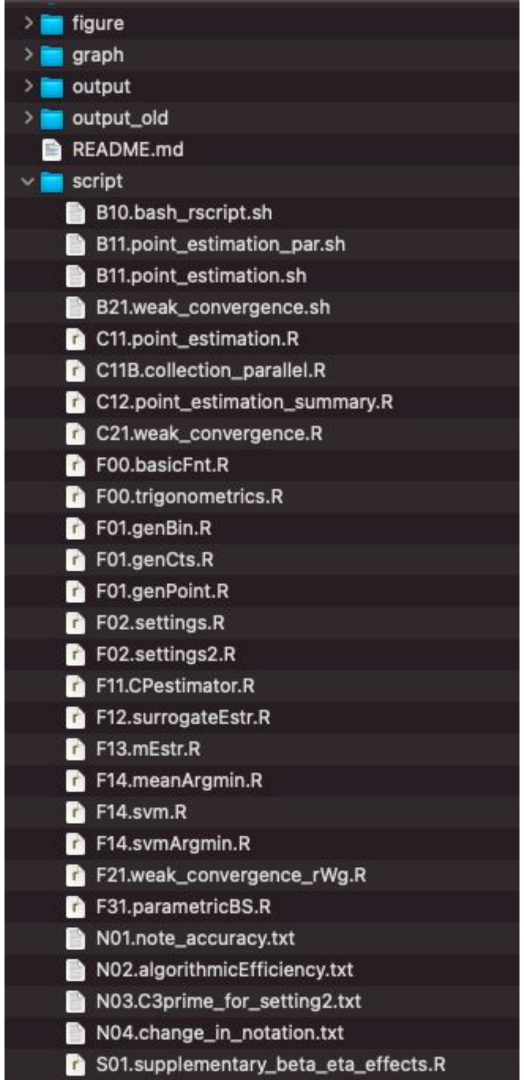
```
1-1.locf.r                                   Jul 1, 201
1-2.mixed_schiz (full data).html             Apr 8, 20
1-2.mixed_schiz.pdf                          Apr 8, 20
1-2.mixed_schiz.rmd                          Apr 8, 20
1-3.mixed_CRP.html                           Mar 21, 2
1-3.mixed_CRP.Rmd                            Jan 10, 2
1-4.mixed_CRP_long_over20removed.html        Mar 26, 2
1-4.mixed_CRP_long.html                      Apr 8, 20
1-4.mixed_CRP_long.pdf                       Mar 25, 2
1-4.mixed_CRP_long.Rmd                       Aug 5, 20
```



```
1-1.locf.r                                   Jul 1, 201
1-2.mixed_schiz (full data).html             Apr 8, 20
1-2.mixed_schiz.pdf                          Apr 8, 20
1-2.mixed_schiz.rmd                          Apr 8, 20
1-3.mixed_CRP.html                           Mar 21, 2
1-3.mixed_CRP.Rmd                            Jan 10, 2
1-4.mixed_CRP_long_over20removed.html        Mar 26, 2
1-4.mixed_CRP_long.html                      Apr 8, 20
1-4.mixed_CRP_long.pdf                       Mar 25, 2
1-4.mixed_CRP_long.Rmd                       Aug 5, 20
```



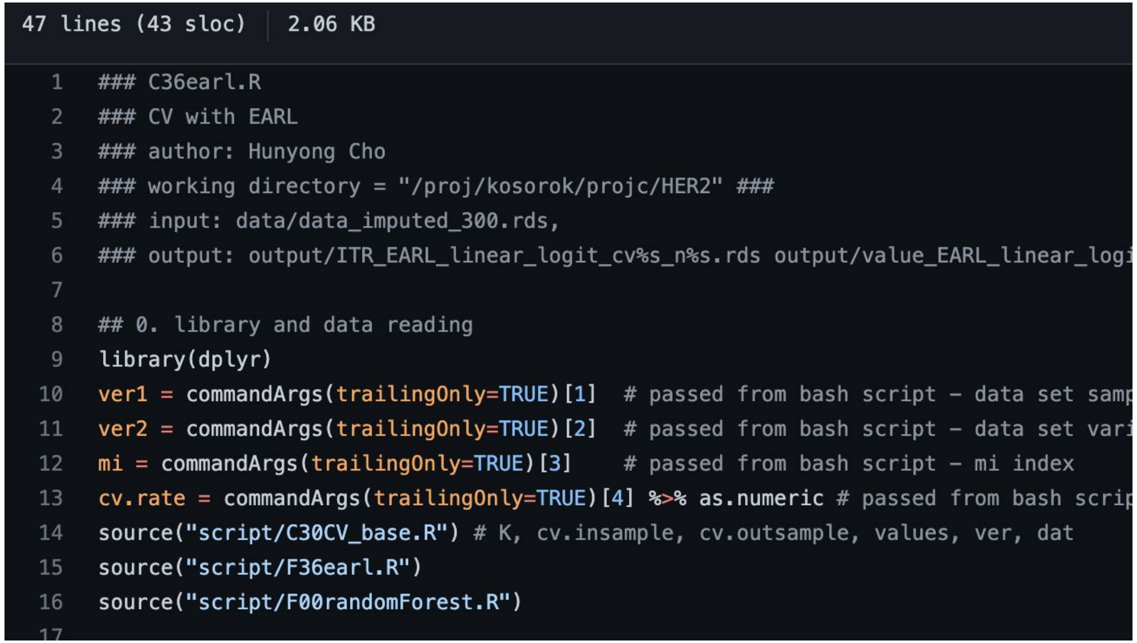
```
  datalist.rds                               Mar 14, 20
  fig2.lowess_log.html                       Feb 28, 20
> figure_decisionlist                        Nov 5, 202
> figure_decisionlist_base_rawdata           Nov 5, 202
  figure_decisionlist_base_rawdata.zip       Jul 16, 202
  fit1.rds                                   Mar 4, 201
  fit2.rds                                   Mar 4, 201
  fit3.rds                                   Mar 5, 201
> graphics                                   Apr 21, 201
  lowess_log.html                            Feb 28, 20
```



```
> figure
> graph
> output
> output_old
  README.md
v script
    B10.bash_rscript.sh
    B11.point_estimation_par.sh
    B11.point_estimation.sh
    B21.weak_convergence.sh
    C11.point_estimation.R
    C11B.collection_parallel.R
    C12.point_estimation_summary.R
    C21.weak_convergence.R
    F00.basicFnt.R
    F00.trigonometrics.R
    F01.genBin.R
    F01.genCts.R
    F01.genPoint.R
    F02.settings.R
    F02.settings2.R
    F11.CPestimator.R
    F12.surrogateEstr.R
    F13.mEstr.R
    F14.meanArgmin.R
    F14.svm.R
    F14.svmArgmin.R
    F21.weak_convergence_rWg.R
    F31.parametricBS.R
    N01.note_accuracy.txt
    N02.algorithmicEfficiency.txt
    N03.C3prime_for_setting2.txt
    N04.change_in_notation.txt
    S01.supplementary_beta_eta_effects.R
```

- Control of access
  (read-only input files, editable script files, output files)
- Flow of work process, dependency
  (data step, preprocessing step, implementation, analysis and visualization)
- Collaboration
  (ownership)

Headers for a team project - overview & ownership

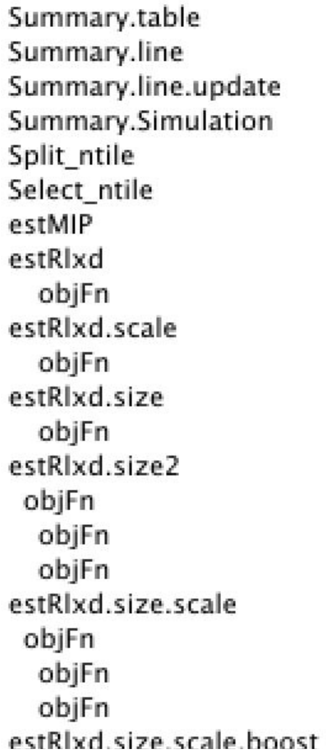


```
47 lines (43 sloc) | 2.06 KB
```

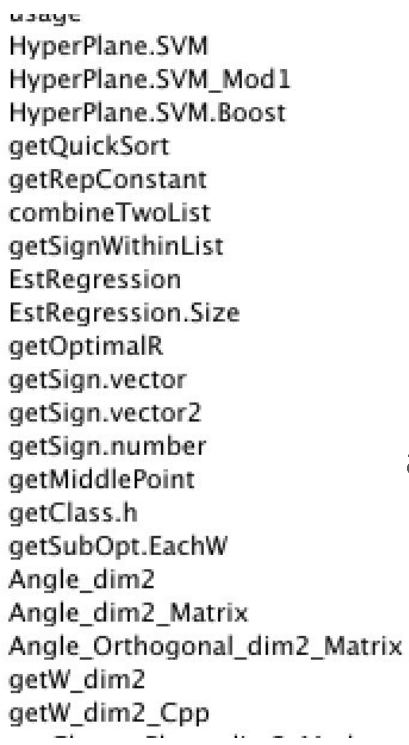
```r
### C36earl.R
### CV with EARL
### author: Hunyong Cho
### working directory = "/proj/kosorok/projc/HER2" ###
### input: data/data_imputed_300.rds,
### output: output/ITR_EARL_linear_logit_cv%s_n%s.rds output/value_EARL_linear_logi

## 0. library and data reading
library(dplyr)
ver1 = commandArgs(trailingOnly=TRUE)[1]  # passed from bash script - data set samp
ver2 = commandArgs(trailingOnly=TRUE)[2]  # passed from bash script - data set vari
mi = commandArgs(trailingOnly=TRUE)[3]    # passed from bash script - mi index
cv.rate = commandArgs(trailingOnly=TRUE)[4] %>% as.numeric # passed from bash scrip
source("script/C30CV_base.R") # K, cv.insample, cv.outsample, values, ver, dat
source("script/F36earl.R")
source("script/F00randomForest.R")
```

Cancer project
https://github.com/KosorokLab/HER2/blob/master/script/C36earl.R

Functions within a file



```
Summary.table
Summary.line
Summary.line.update
Summary.Simulation
Split_ntile
Select_ntile
estMIP
estRlxd
   objFn
estRlxd.scale
   objFn
estRlxd.size
   objFn
estRlxd.size2
  objFn
   objFn
   objFn
estRlxd.size.scale
  objFn
   objFn
   objFn
estRlxd.size.scale.boost
```



```
usage
HyperPlane.SVM
HyperPlane.SVM_Mod1
HyperPlane.SVM.Boost
getQuickSort
getRepConstant
combineTwoList
getSignWithinList
EstRegression
EstRegression.Size
getOptimalR
getSign.vector
getSign.vector2
getSign.number
getMiddlePoint
getClass.h
getSubOpt.EachW
Angle_dim2
Angle_dim2_Matrix
Angle_Orthogonal_dim2_Matrix
getW_dim2
getW_dim2_Cpp
```

and more ...

What can be improved?

- Too many functions / code in a file
- No structure (no clue about what function is used for what)
- Redundancy (arguments in functions, version control tools)

Solutions

- Split a file into a meaningful units: task, functionality
- Simple but meaningful names
- Utilize arguments in functions,
- Remove the unused and use version control (git)

Style guides

Google Style Guides https://google.github.io/styleguide/

R, C, python, shell, ...

Hadley Wickham's Style Guides http://stat405.had.co.nz/r-style.html

Jenny Bryan’s Style Guides
https://www.stat.ubc.ca/~jenny/STAT545A/block19_codeFormattingOrganization.html

“please TRUST ME when I say that your coding style is very, very important to the quality of work and your happiness in it.” - Jenny Bryan @ Rstudio & UBC

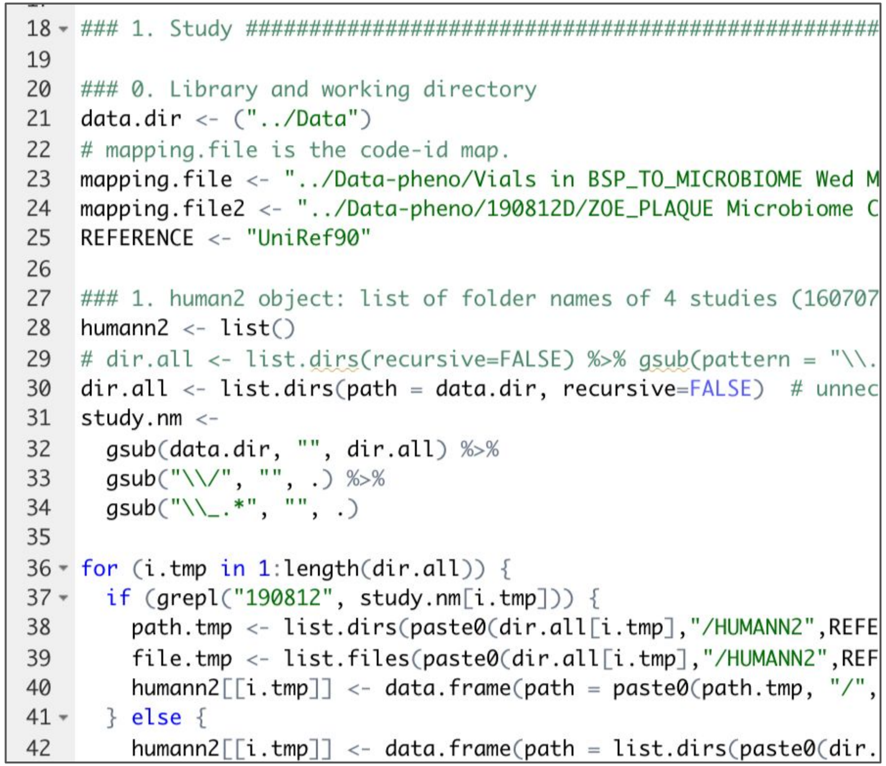
What do you do if you are midstream in the project and want to bring some organization?

1. Make a backup copy so that you can always go back
2. Set up a git repository for version control
3. Draw a workflow, establish a folder structure.
4. Revision: move/rename files or create files from scratch according to the order.
5. Validation: validate the code for each function / file.

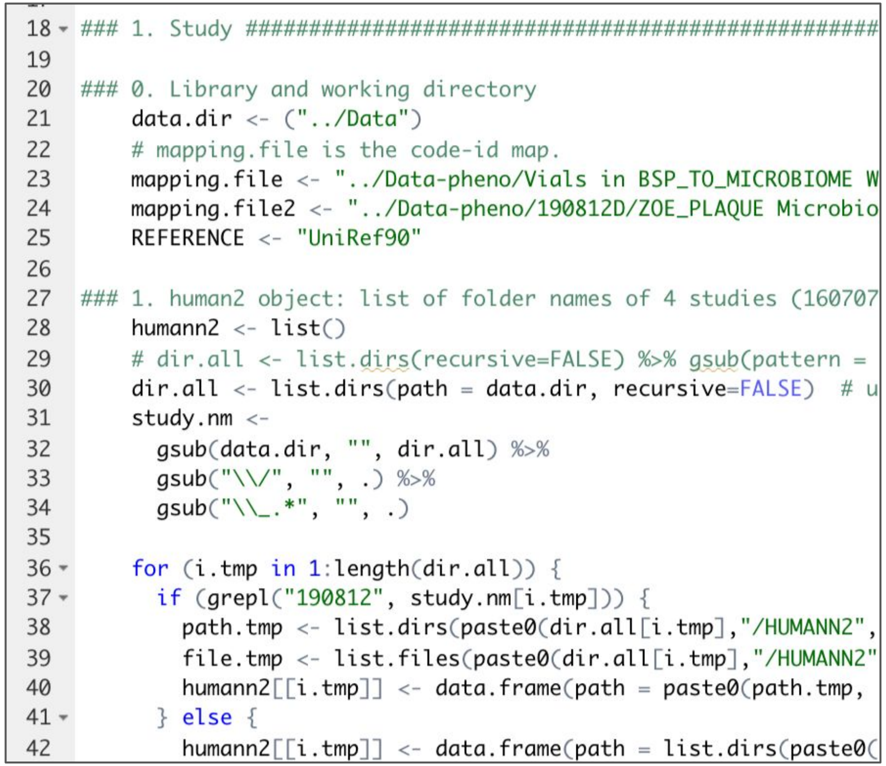
Some coding tips in R

Some tips in R - for readability

indentation



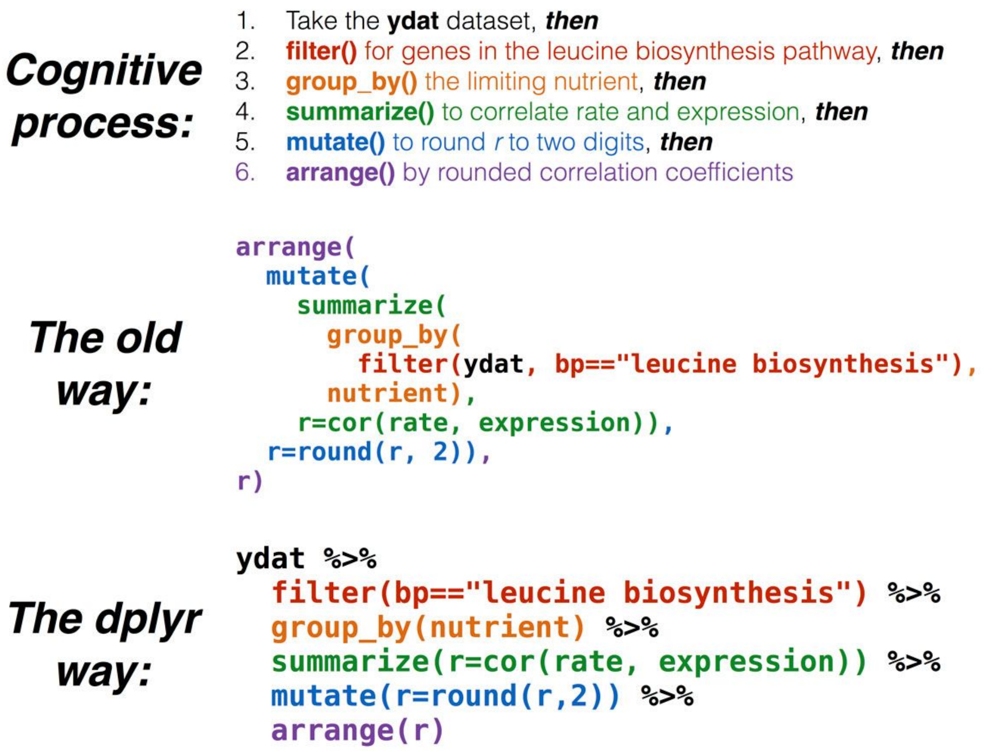
```
### 1. Study ##################################################

### 0. Library and working directory
data.dir <- ("../Data")
# mapping.file is the code-id map.
mapping.file <- "../Data-pheno/Vials in BSP_TO_MICROBIOME Wed M
mapping.file2 <- "../Data-pheno/190812D/ZOE_PLAQUE Microbiome C
REFERENCE <- "UniRef90"

### 1. human2 object: list of folder names of 4 studies (160707
humann2 <- list()
# dir.all <- list.dirs(recursive=FALSE) %>% gsub(pattern = "\\.
dir.all <- list.dirs(path = data.dir, recursive=FALSE)  # unnec
study.nm <-
  gsub(data.dir, "", dir.all) %>%
  gsub("\\/", "", .) %>%
  gsub("\\_.*", "", .)

for (i.tmp in 1:length(dir.all)) {
  if (grepl("190812", study.nm[i.tmp])) {
    path.tmp <- list.dirs(paste0(dir.all[i.tmp],"/HUMANN2",REFE
    file.tmp <- list.files(paste0(dir.all[i.tmp],"/HUMANN2",REF
    humann2[[i.tmp]] <- data.frame(path = paste0(path.tmp, "/",
  } else {
    humann2[[i.tmp]] <- data.frame(path = list.dirs(paste0(dir.
```



```
### 1. Study ##################################################

### 0. Library and working directory
    data.dir <- ("../Data")
    # mapping.file is the code-id map.
    mapping.file <- "../Data-pheno/Vials in BSP_TO_MICROBIOME W
    mapping.file2 <- "../Data-pheno/190812D/ZOE_PLAQUE Microbio
    REFERENCE <- "UniRef90"

### 1. human2 object: list of folder names of 4 studies (160707
    humann2 <- list()
    # dir.all <- list.dirs(recursive=FALSE) %>% gsub(pattern =
    dir.all <- list.dirs(path = data.dir, recursive=FALSE)  # u
    study.nm <-
      gsub(data.dir, "", dir.all) %>%
      gsub("\\/", "", .) %>%
      gsub("\\_.*", "", .)

    for (i.tmp in 1:length(dir.all)) {
      if (grepl("190812", study.nm[i.tmp])) {
        path.tmp <- list.dirs(paste0(dir.all[i.tmp],"/HUMANN2",
        file.tmp <- list.files(paste0(dir.all[i.tmp],"/HUMANN2"
        humann2[[i.tmp]] <- data.frame(path = paste0(path.tmp,
      } else {
        humann2[[i.tmp]] <- data.frame(path = list.dirs(paste0(
```

Some tips in R - for readability

piping (R:dplyr)



Cognitive process:

1. Take the **ydat** dataset, ***then***
2. **filter()** for genes in the leucine biosynthesis pathway, ***then***
3. **group_by()** the limiting nutrient, ***then***
4. **summarize()** to correlate rate and expression, ***then***
5. **mutate()** to round *r* to two digits, ***then***
6. **arrange()** by rounded correlation coefficients

The old way:

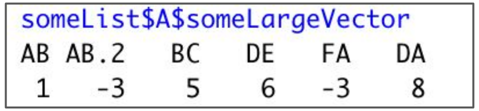
```
arrange(
  mutate(
    summarize(
      group_by(
        filter(ydat, bp=="leucine biosynthesis"),
      nutrient),
    r=cor(rate, expression)),
  r=round(r, 2)),
r)
```

The dplyr way:

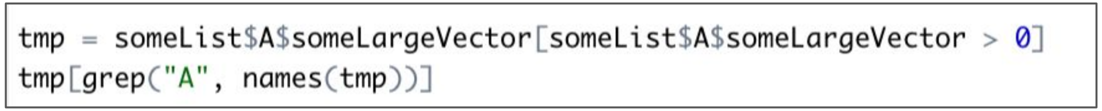
```
ydat %>%
  filter(bp=="leucine biosynthesis") %>%
  group_by(nutrient) %>%
  summarize(r=cor(rate, expression)) %>%
  mutate(r=round(r,2)) %>%
  arrange(r)
```

Some tips in R - for readability

piping

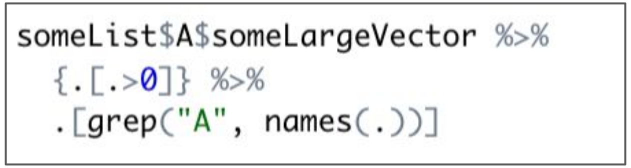


```
someList$A$someLargeVector
AB  AB.2    BC    DE    FA    DA
 1    -3     5     6    -3     8
```



```
tmp = someList$A$someLargeVector[someList$A$someLargeVector > 0]
tmp[grep("A", names(tmp))]
```

vs



```
someList$A$someLargeVector %>%
  {.[.>0]} %>%
  .[grep("A", names(.))]
```

Some tips in R - for readability

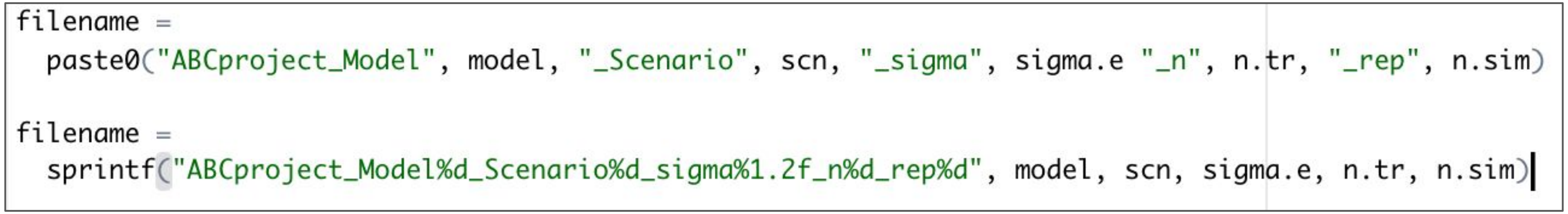
tidyverse

tibble	data tables with a nicer interface
dplyr	piping
tidyr	reshaping
stringr	strings manipulation
purrr	functional programming
...	

https://www.tidyverse.org/
Phoebe Jiang's workshop (Lab drive > Presentations > dplyr workshop)

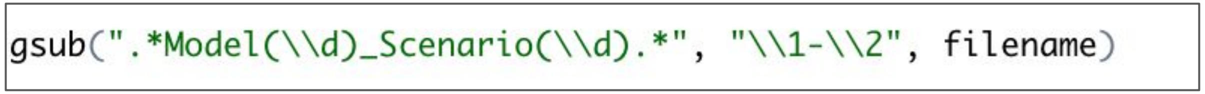
Some tips in R

paste vs sprintf



```
filename =
  paste0("ABCproject_Model", model, "_Scenario", scn, "_sigma", sigma.e "_n", n.tr, "_rep", n.sim)

filename =
  sprintf("ABCproject_Model%d_Scenario%d_sigma%1.2f_n%d_rep%d", model, scn, sigma.e, n.tr, n.sim)
```

regex

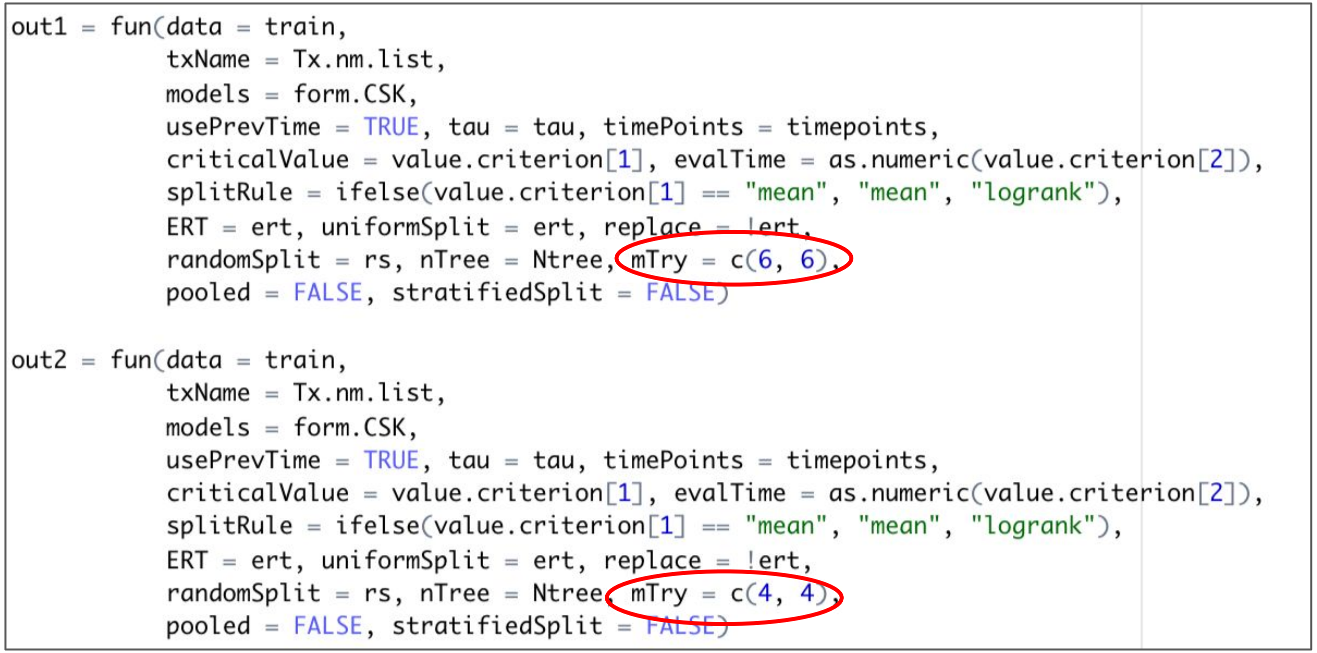


```
gsub(".*Model(\\d)_Scenario(\\d).*", "\\1-\\2", filename)
```

Basic regex syntax: http://www.endmemo.com/r/grep.php

Some tips in R - dry coding

repetition of function arguments

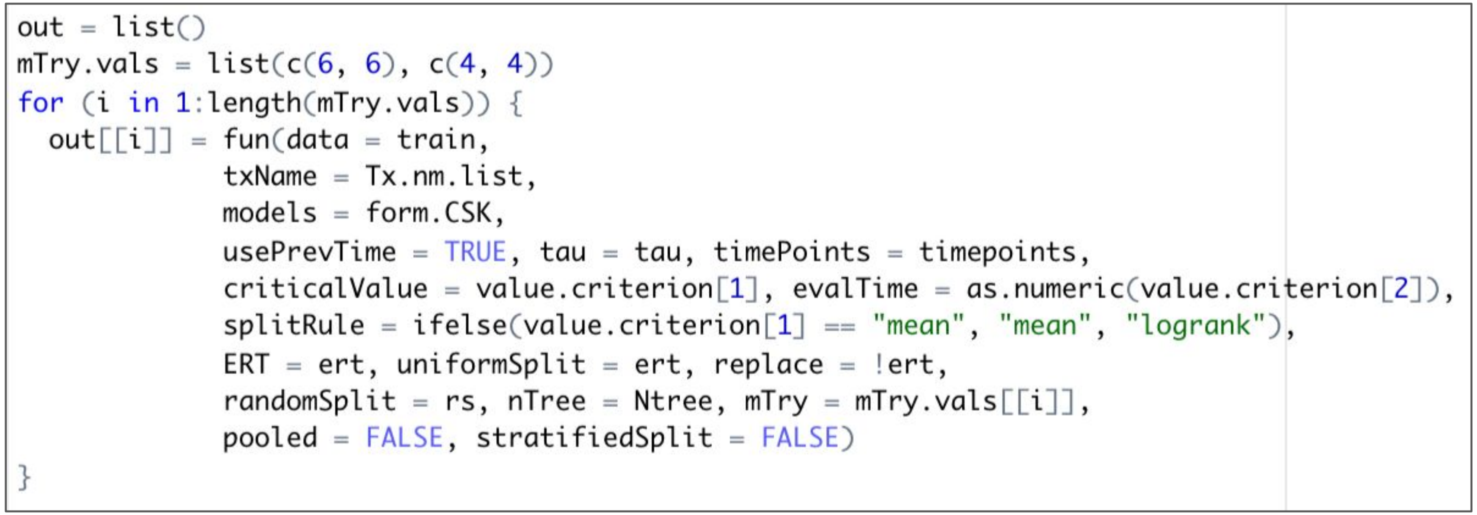


```
out1 = fun(data = train,
           txName = Tx.nm.list,
           models = form.CSK,
           usePrevTime = TRUE, tau = tau, timePoints = timepoints,
           criticalValue = value.criterion[1], evalTime = as.numeric(value.criterion[2]),
           splitRule = ifelse(value.criterion[1] == "mean", "mean", "logrank"),
           ERT = ert, uniformSplit = ert, replace = !ert,
           randomSplit = rs, nTree = Ntree, mTry = c(6, 6),
           pooled = FALSE, stratifiedSplit = FALSE)

out2 = fun(data = train,
           txName = Tx.nm.list,
           models = form.CSK,
           usePrevTime = TRUE, tau = tau, timePoints = timepoints,
           criticalValue = value.criterion[1], evalTime = as.numeric(value.criterion[2]),
           splitRule = ifelse(value.criterion[1] == "mean", "mean", "logrank"),
           ERT = ert, uniformSplit = ert, replace = !ert,
           randomSplit = rs, nTree = Ntree, mTry = c(4, 4),
           pooled = FALSE, stratifiedSplit = FALSE)
```

...

Some tips in R - dry coding

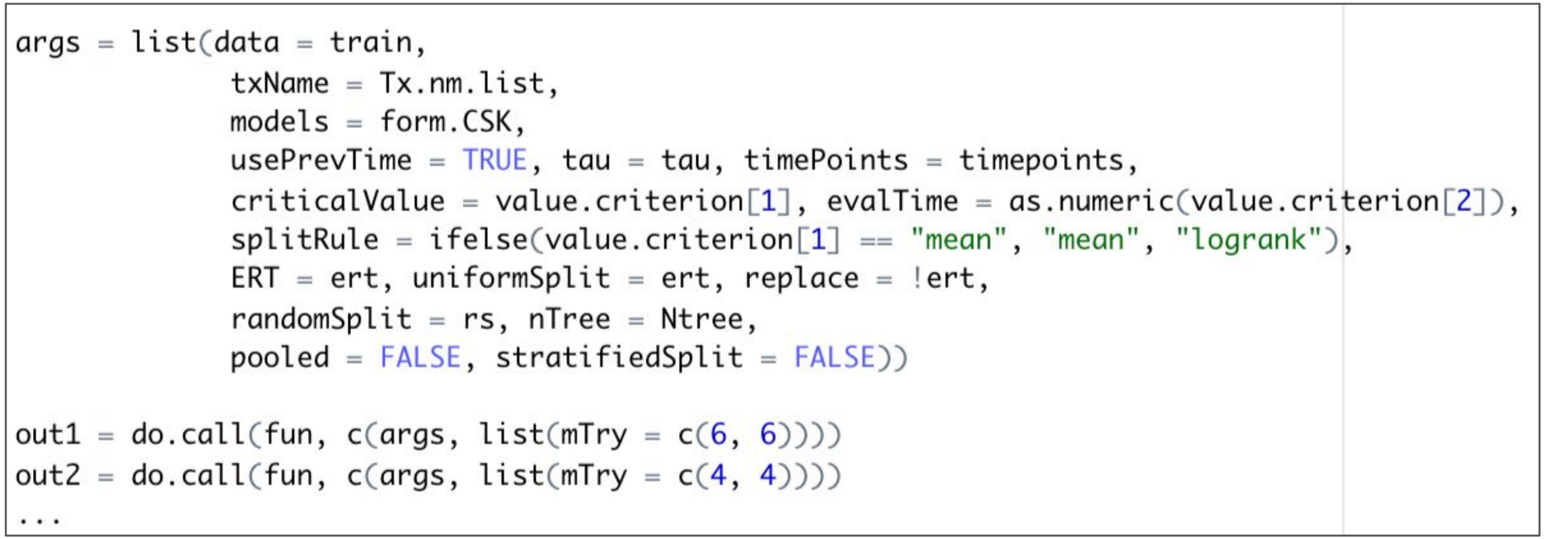
repetition of function arguments - Solution 1: for loop



```r
out = list()
mTry.vals = list(c(6, 6), c(4, 4))
for (i in 1:length(mTry.vals)) {
  out[[i]] = fun(data = train,
              txName = Tx.nm.list,
              models = form.CSK,
              usePrevTime = TRUE, tau = tau, timePoints = timepoints,
              criticalValue = value.criterion[1], evalTime = as.numeric(value.criterion[2]),
              splitRule = ifelse(value.criterion[1] == "mean", "mean", "logrank"),
              ERT = ert, uniformSplit = ert, replace = !ert,
              randomSplit = rs, nTree = Ntree, mTry = mTry.vals[[i]],
              pooled = FALSE, stratifiedSplit = FALSE)
}
```

Some tips in R - dry coding

repetition of function arguments - Solution 2: do.call()



```r
args = list(data = train,
            txName = Tx.nm.list,
            models = form.CSK,
            usePrevTime = TRUE, tau = tau, timePoints = timepoints,
            criticalValue = value.criterion[1], evalTime = as.numeric(value.criterion[2]),
            splitRule = ifelse(value.criterion[1] == "mean", "mean", "logrank"),
            ERT = ert, uniformSplit = ert, replace = !ert,
            randomSplit = rs, nTree = Ntree,
            pooled = FALSE, stratifiedSplit = FALSE))

out1 = do.call(fun, c(args, list(mTry = c(6, 6))))
out2 = do.call(fun, c(args, list(mTry = c(4, 4))))
...
```

Some tips in R - dry coding

Excessive use of ifelse - alternatives

```
param =
  if (opt == "A") {
    1
  } else if (opt == "B") {
    3
  } else if (opt == "C") {
    8
  } else if (opt == "D") {
    10
  }
```

Alternative 1: dictionary, a named vector or list

```
params = c(A = 1, B = 3, C = 8, D = 10)
param = params[opt]
```

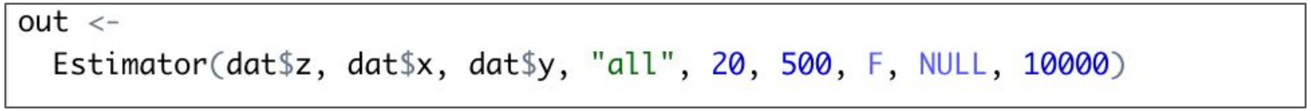
Alternative 2: switch()

```
param = switch(opt, A = 1, B = 3, C = 8, D = 10)
```

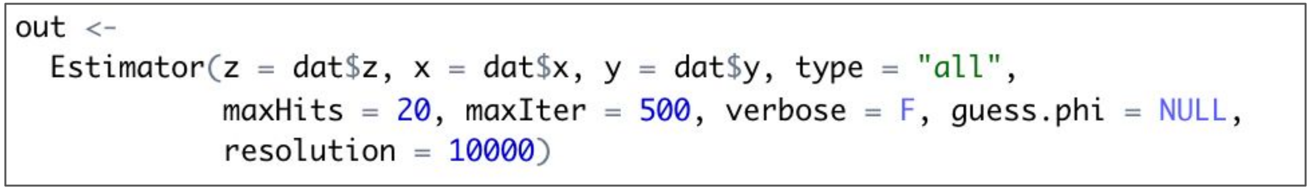
Some tips in R - validity

Some habits for preventing leaks

1. **explicitly add argument names**



```r
out <-
  Estimator(dat$z, dat$x, dat$y, "all", 20, 500, F, NULL, 10000)
```

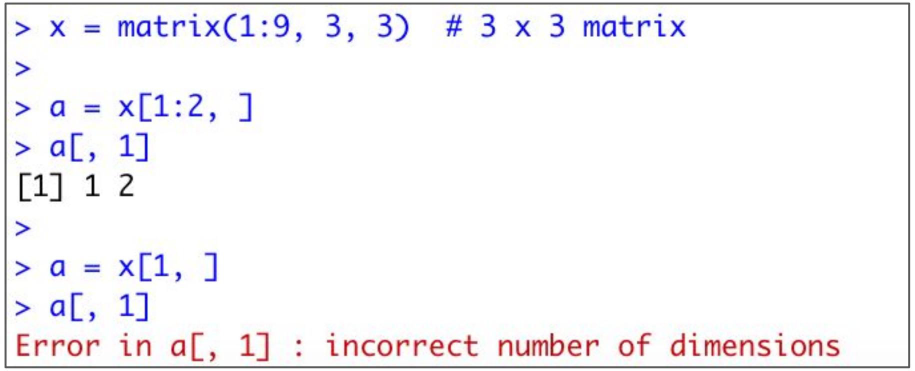


```r
out <-
  Estimator(z = dat$z, x = dat$x, y = dat$y, type = "all",
            maxHits = 20, maxIter = 500, verbose = F, guess.phi = NULL,
            resolution = 10000)
```

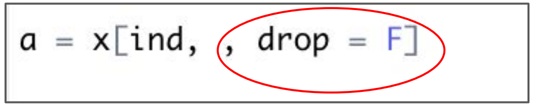
Some tips in R - validity

Some habits for preventing leaks

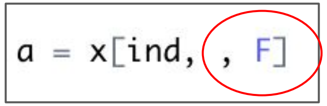
2. drop = FALSE in matrix/array subsetting



```
> x = matrix(1:9, 3, 3)  # 3 x 3 matrix
>
> a = x[1:2, ]
> a[, 1]
[1] 1 2
>
> a = x[1, ]
> a[, 1]
Error in a[, 1] : incorrect number of dimensions
```



```
a = x[ind, , drop = F]
```



```
a = x[ind, , F]
```

Some tips in *Rstudio*

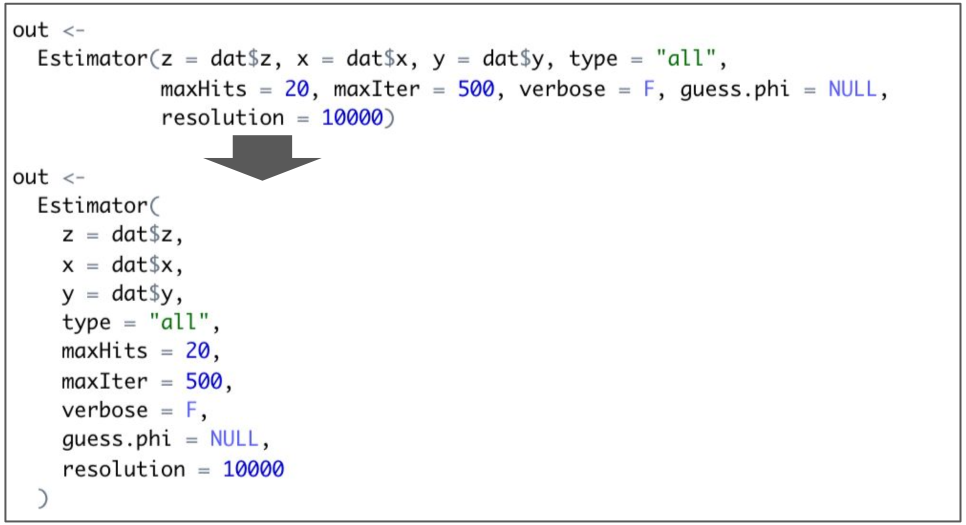
Global search

Control (Cmd) + shift + F
When you want to find something, but not know what file it is

Tidying up code

Control (Cmd) + shift + A
(after selection)

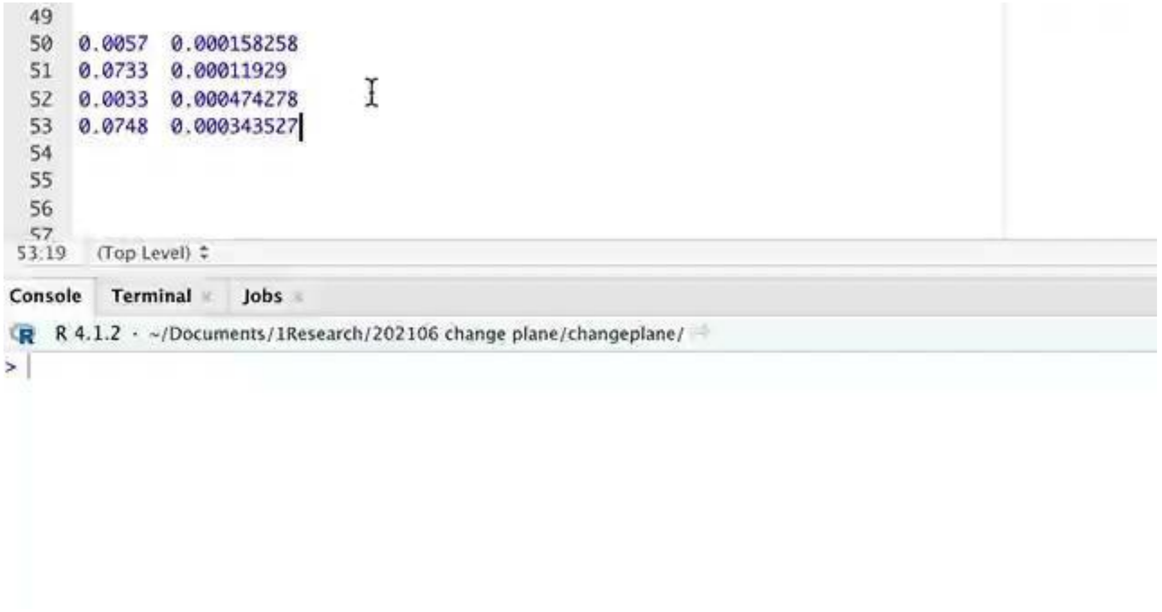
Reformatting



```
out <-
  Estimator(z = dat$z, x = dat$x, y = dat$y, type = "all",
            maxHits = 20, maxIter = 500, verbose = F, guess.phi = NULL,
            resolution = 10000)
```

```
out <-
  Estimator(
    z = dat$z,
    x = dat$x,
    y = dat$y,
    type = "all",
    maxHits = 20,
    maxIter = 500,
    verbose = F,
    guess.phi = NULL,
    resolution = 10000
  )
```

Multiple cursors (Rstudio, Overleaf, a bunch of IDEs)



```
0.0057  0.000158258
0.0733  0.00011929
0.0033  0.000474278
0.0748  0.000343527
```

53:19
(Top Level)
Console
Terminal
Jobs
R 4.1.2 · ~/Documents/1Research/202106 change plane/changeplane/

Alt (opt) + drag

Useful for table editing in latex!