drake

Reproducibility and high-performance computing

Will Landau, Eli Lilly and Company

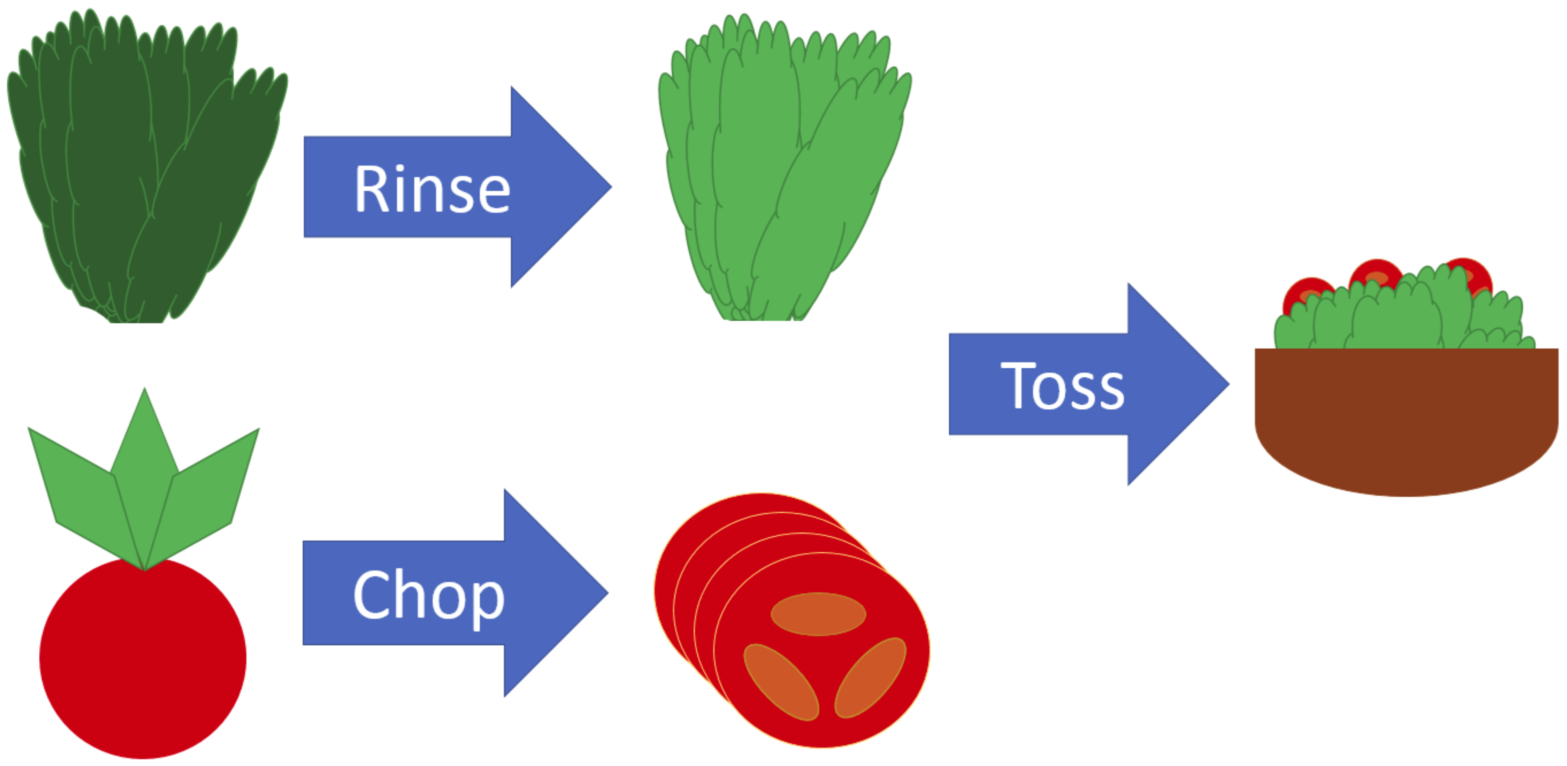
Data Frames in R for Make

1. Save time
2. Stay reproducible
3. Organize your project

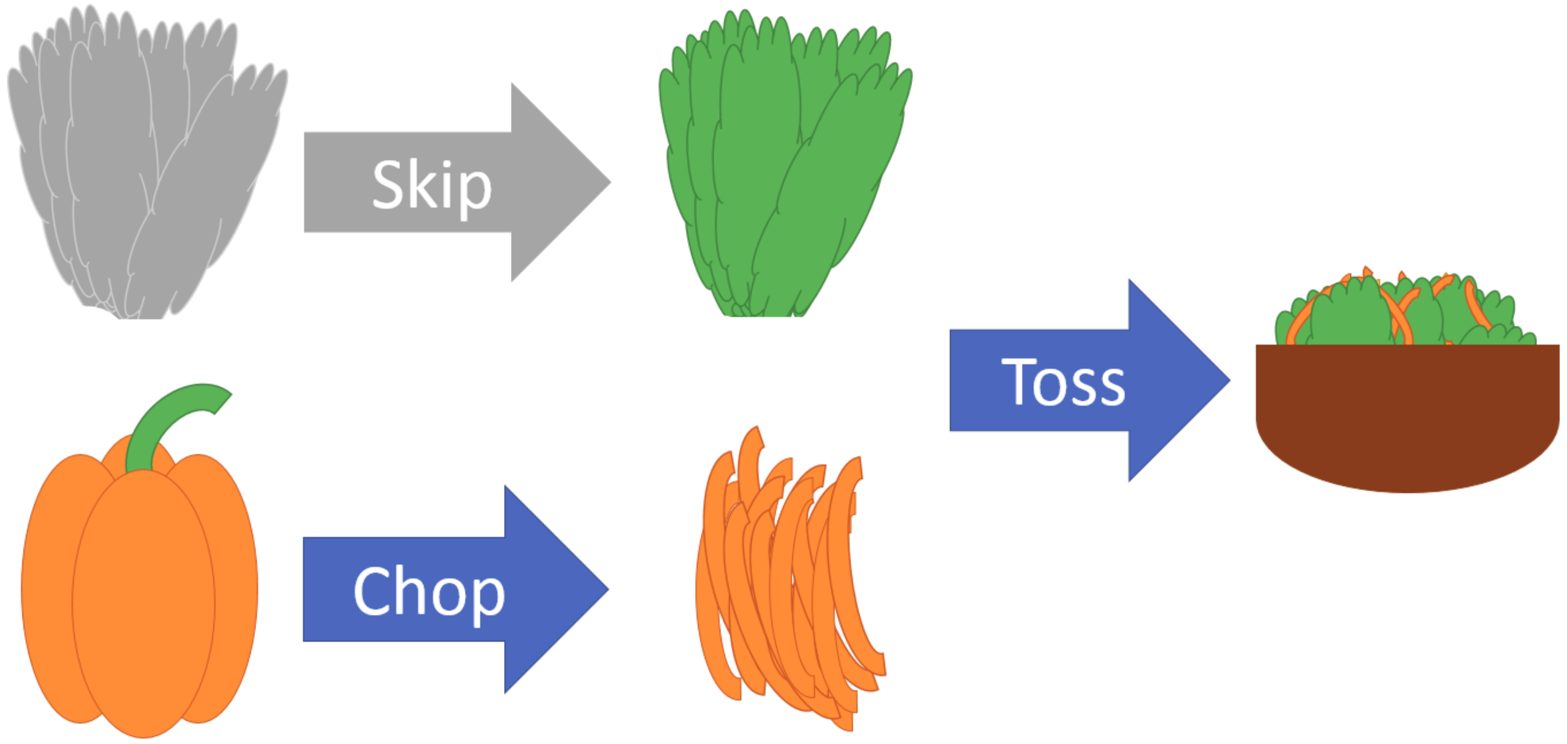
From a colleague

"The fastest code is the code you do not run."

Kirill Müller's cooking analogy



Change recipe and reuse leftovers



Plan a project

Your plan is a data frame

```
whole_plan
```

```
##                  target                                  command
## 1           'report.md' my_knit('report.Rmd', report_dependencies)
## 2 report_dependencies          c(small, large, regression2_small)
## 3                 small                              simulate(5)
## 4                 large                             simulate(50)
## 5     regression1_small                              reg1(small)
## 6     regression1_large                              reg1(large)
## 7     regression2_small                              reg2(small)
## 8     regression2_large                              reg2(large)
```

Minimize typing

```
methods
```

```
##         target              command
## 1 regression1 reg1(..dataset..)
## 2 regression2 reg2(..dataset..)
```

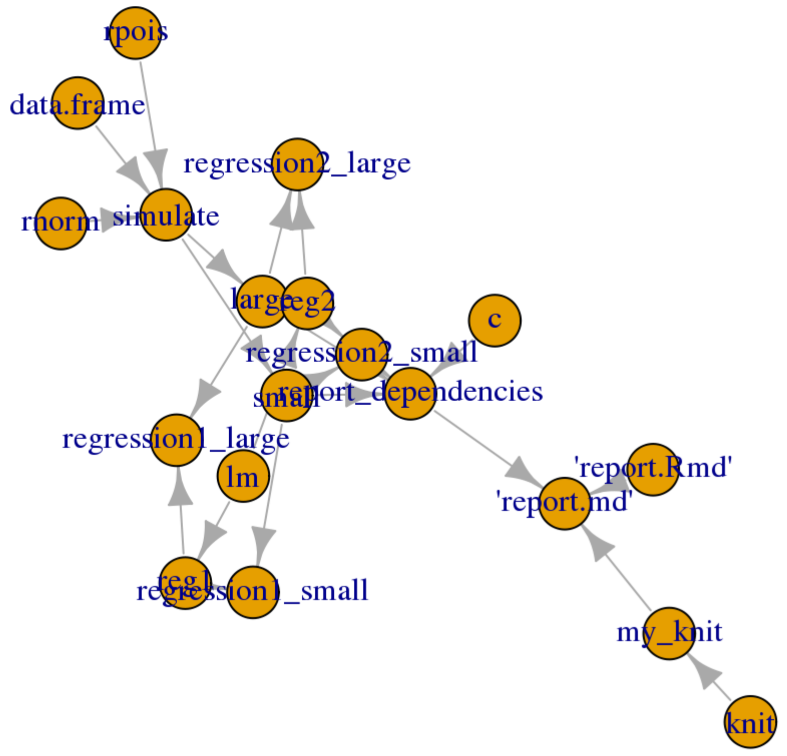
```
analyses(plan = methods, datasets = data_plan)
```

```
##               target     command
## 1 regression1_small reg1(small)
## 2 regression1_large reg1(large)
## 3 regression2_small reg2(small)
## 4 regression2_large reg2(large)
```

Minimize typing

- `plan()`
- `analyses()`
- `summaries()`
- `expand()`
- `evaluate()`
- `gather()`

The dependency graph



The dependency graph

- `plot_graph()`
- `build_graph()` (igraph object)
- `read_graph()` (run project first)
- `tracked()` (just list the nodes)

Run the project

Run the project

```
make(whole_plan)
```

```
## import 'report.Rmd'
## import c
## import knit
## import data.frame
## import rnorm
## import rpois
## import lm
## import my_knit
## import simulate
## import reg1
## import reg2
## build small
## build large
## build regression1_small
## build regression1_large
## build regression2_small
## build regression2_large
## build report_dependencies
## build 'report.md'
```

What did you make? How did it go?

```r
status() # see also: session()
```

```
##          'report.md'         'report.Rmd'                    c 
##           "finished"           "finished"           "finished" 
##           data.frame                 knit                large 
##           "finished"           "finished"           "finished" 
##                   lm              my_knit                 reg1 
##           "finished"           "finished"           "finished" 
##                 reg2    regression1_large    regression1_small 
##           "finished"           "finished"           "finished" 
##    regression2_large    regression2_small  report_dependencies 
##           "finished"           "finished"           "finished" 
##                rnorm                rpois             simulate 
##           "finished"           "finished"           "finished" 
##                small 
##           "finished"
```

Reproducibility: the results match the code

```
make(whole_plan)
```

```
## Unloading targets from environment:
##   report_dependencies
## import 'report.Rmd'
## import c
## import knit
## import data.frame
## import rnorm
## import rpois
## import lm
## import my_knit
## import simulate
## import reg1
## import reg2
```

Reproducibility: the results match the code

```
status() # Set imported_files_only=TRUE to ignore imported non-files.
```

```
## 'report.Rmd'            c   data.frame         knit           lm 
##   "finished"   "finished"   "finished"   "finished"   "finished" 
##      my_knit         reg1         reg2        rnorm        rpois 
##   "finished"   "finished"   "finished"   "finished"   "finished" 
##     simulate 
##   "finished"
```

Access the output

```
head(cached()) # some of the cached objects
```

```
## [1] "'report.md'"  "'report.Rmd'" "c"              "data.frame"
## [5] "knit"         "large"
```

```
readd(small) # loadd(small, large)
```

```
##            x y
## 1  1.3237380 0
## 2 -1.3729077 0
## 3  0.4513560 1
## 4  1.4094745 2
## 5 -0.7707691 2
```

Change an ingredient

```
reg2
```

```
## function(d){
##   d$x2 = d$x^2
##   lm(y ~ x2, data = d)
## }
```

```
reg2 = function(d){
  d$x3 = d$x^3 # new cubic term
  lm(y ~ x3, data = d)
}
```

Reuse your leftovers

```r
make(whole_plan)
```

```
## import 'report.Rmd'
## import c
## import knit
## import data.frame
## import rnorm
## import rpois
## import lm
## import my_knit
## import simulate
## import reg1
## import reg2
## build regression2_small
## build regression2_large
## build report_dependencies
## build 'report.md'
```

Reuse your leftovers

```r
status(imported_files_only = TRUE)
```

```
##          'report.md'        'report.Rmd'    regression2_large
##           "finished"          "finished"           "finished"
##    regression2_small report_dependencies
##           "finished"          "finished"
```

High-performance computing

Auto-magically switch on parallel computing

- Parallel processes (multiple chefs)

```
make(whole_plan, jobs = 2) # Backend chosen based on platform.
make(whole_plan, parallelism = "mclapply", jobs = 2) # Mac/Linux
make(whole_plan, parallelism = "parLapply", jobs = 2) # Windows too
```

- Parallel R sessions (multiple kitchens)

```
make(whole_plan, parallelism = "Makefile", jobs = 2)
make(whole_plan, parallelism = "Makefile", command = "make",
     args = c("--jobs=2", "--silent"))
```

Supercomputing

`my_script.R`

```r
...
make(whole_plan, parallelism = "Makefile", jobs = 8,
  prepend = "SHELL = ./shell.sh")
```

`shell.sh`

```bash
#!/bin/bash
shift
echo "module load R; $*" | qsub -sync y -cwd -j y
```

Run on a cluster or supercomputer.

```bash
chmod +x shell.sh
nohup nice -19 R CMD BATCH my_script.R &
```

Try it out

```
install.packages("drake")
library(drake)
example_drake("basic") # Write example code to try.
vignette(package = "drake") # List the vignettes.
vignette("drake") # high-level overview
vignette("quickstart") # annotated example
vignette("caution") # avoid pitfalls
```

- Rendered vignettes:
- https://CRAN.R-project.org/package=drake/vignettes
- Bug reports, issues, feature requests:
- https://github.com/wlandau-lilly/drake/issues

Inspiration

- Huge inspiration: the remake package by Rich FitzJohn
  - https://github.com/richfitz/remake
- GNU Make
  - https://www.gnu.org/software/make

Sources

- FitzJohn, Rich. "Remake: Make-like declarative workflows in R." 2017. R package version 0.3.0. GitHub repository, https://github.com/richfitz/remake.
- Landau, William M. "Drake: data frames in R for Make." R package version 3.0.0. https://CRAN.R-project.org/package=drake.
- Müller, Kirill. "Reproducible workflows with R." Zurich R user meetup. April 10, 2017. https://krlmlr.github.io/remake-slides.
- Stallman, Richard M. and McGrath, Roland and Smith, Paul D. GNU Make: A Program for Directing Recompilation, for version 3.81. Free Software Foundation, 2004.