(A bit of) Advanced R

Efficient R programming

Julien Chiquet

Université Paris Dauphine

Juin 2018

http://github/jchiquet/CourseAdvancedR

Resources

- Gillespie & Lovelace (2016): efficient R programming
- Wickham (2014)

Part 0: Prerequesties

- xpply family, do.call, Reduce

Outline

Quick (and dirty) benchmarking with `system.time()`

One usually relies on the command `system.time(expr)` to evaluate the timings:

```
func.one <- function(n) {return(rnorm(n,0,1))}
func.two <- function(n) {return(rpois(n,1))}

n <- 1000
system.time(replicate(100, func.one(n)))
```

```
##    user  system elapsed
##   0.008   0.000   0.010
```

```
system.time(replicate(100, func.two(n)))
```

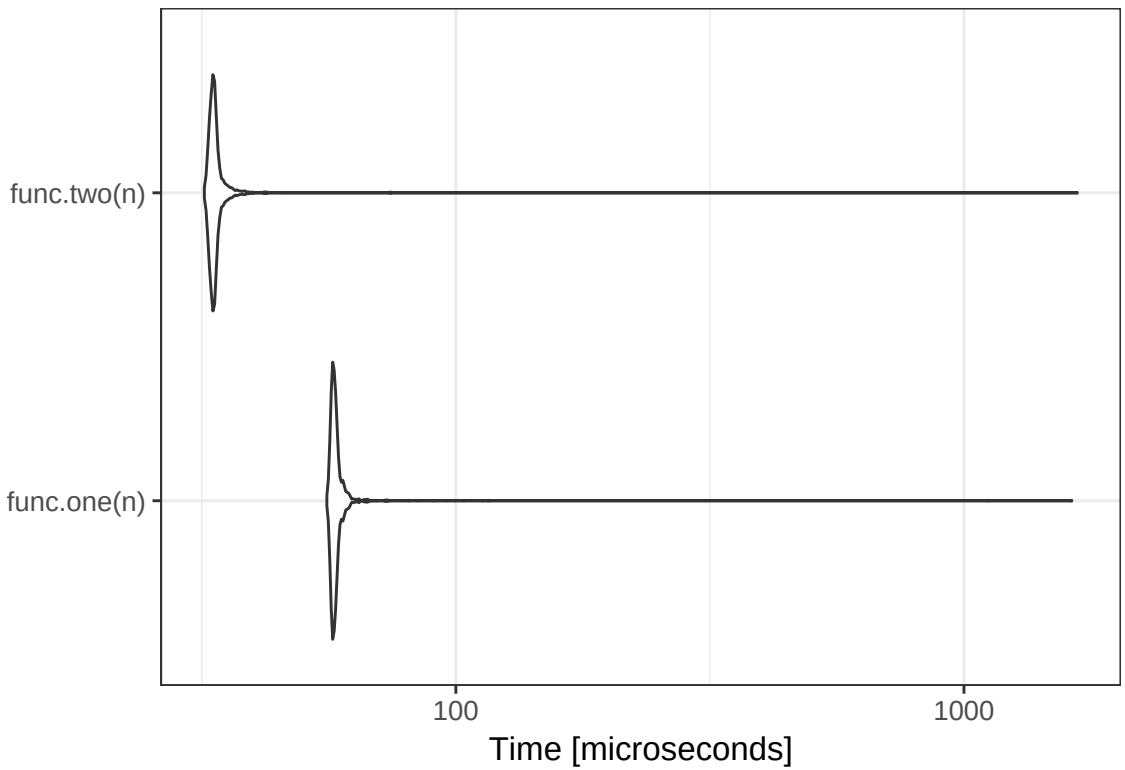
```
##    user  system elapsed
##   0.008   0.000   0.007
```

Quick benchmarking with `microbenchmark`

```
func.one <- function(n) {return(rnorm(n,0,1))}
func.two <- function(n) {return(rpois(n,1))}

library(microbenchmark)

n <- 1000
res <- microbenchmark(func.one(n), func.two(n), times=1000)
ggplot2::autoplot(res)
```



Profile your code with base Rprof I

Suppose you want to evaluate which part of the following function is hot:

```
## generate data, center/scale and perform ridge regression
my_func <- function(n,p) {

  require(MASS)

  ## draw data
  x <- matrix(rnorm(n*p),n,p)
  y <- rnorm(n)

  ## center/scale
  xs <- scale(x)
  ys <- y - mean(y)

  ## return ridge's coefficients
  ridge <- lm.ridge(ys~xs+0,lambda=1)

  return(ridge$coef)
}
```

One can rely on the default Rprof function, with somewhat technical outputs

Profile your code with base Rprof II

```
Rprof(file="profiling.out", interval=0.05)
res <- my_func(1000,500)
Rprof(NULL)
```

```
summaryRprof("profiling.out")$by.self
```

```
##                      self.time self.pct total.time total.pct
## "La.svd"                  0.90    75.00       0.95     79.17
## "matrix"                  0.15    12.50       0.15     12.50
## "FUN"                     0.05     4.17       1.20    100.00
## "aperm.default"           0.05     4.17       0.05      4.17
## "na.omit.data.frame"      0.05     4.17       0.05      4.17
```

```
summaryRprof("profiling.out")$by.total
```

Profile your code with base Rprof III

```
##                         total.time total.pct self.time self.pct
## "FUN"                         1.20    100.00      0.05     4.17
## "block_exec"                  1.20    100.00      0.00     0.00
## "call_block"                  1.20    100.00      0.00     0.00
## "eval"                        1.20    100.00      0.00     0.00
## "evaluate"                    1.20    100.00      0.00     0.00
## "evaluate_call"               1.20    100.00      0.00     0.00
## "evaluate::evaluate"          1.20    100.00      0.00     0.00
## "handle"                      1.20    100.00      0.00     0.00
## "in_dir"                      1.20    100.00      0.00     0.00
## "knit"                        1.20    100.00      0.00     0.00
## "knitr::knit"                 1.20    100.00      0.00     0.00
## "lapply"                      1.20    100.00      0.00     0.00
## "my_func"                     1.20    100.00      0.00     0.00
## "process_file"                1.20    100.00      0.00     0.00
## "process_group"               1.20    100.00      0.00     0.00
## "process_group.block"         1.20    100.00      0.00     0.00
## "rmarkdown::render"           1.20    100.00      0.00     0.00
## "timing_fn"                   1.20    100.00      0.00     0.00
## "withCallingHandlers"         1.20    100.00      0.00     0.00
## "withVisible"                 1.20    100.00      0.00     0.00
## "lm.ridge"                    1.00     83.33      0.00     0.00
## "La.svd"                      0.95     79.17      0.90    75.00
## "svd"                         0.95     79.17      0.00     0.00
## "matrix"                      0.15     12.50      0.15    12.50
## "scale"                       0.10      8.33      0.00     0.00
## "scale.default"               0.10      8.33      0.00     0.00
```

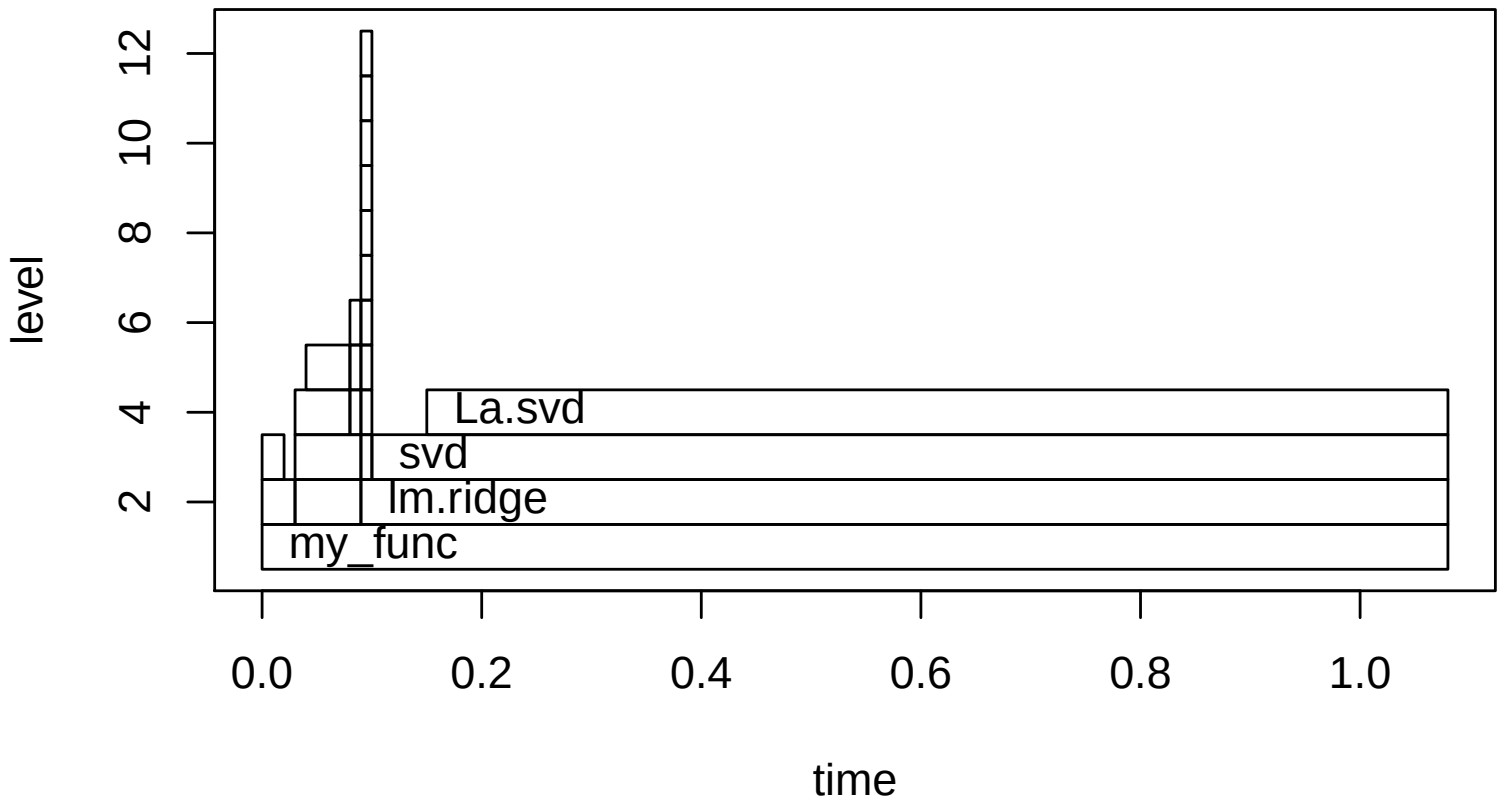
Profile your code with base Rprof IV

```
## "aperm.default"           0.05      4.17      0.05      4.17
## "na.omit.data.frame"      0.05      4.17      0.05      4.17
## "aperm"                   0.05      4.17      0.00      0.00
## "apply"                   0.05      4.17      0.00      0.00
## "eval.parent"             0.05      4.17      0.00      0.00
## ".External2"              0.05      4.17      0.00      0.00
## "model.frame.default"     0.05      4.17      0.00      0.00
## "na.omit"                 0.05      4.17      0.00      0.00
## "stats::model.frame"      0.05      4.17      0.00      0.00
## "sweep"                   0.05      4.17      0.00      0.00
```

Profile your code with `profr`

The *profr* package is maybe a little easier to understand...

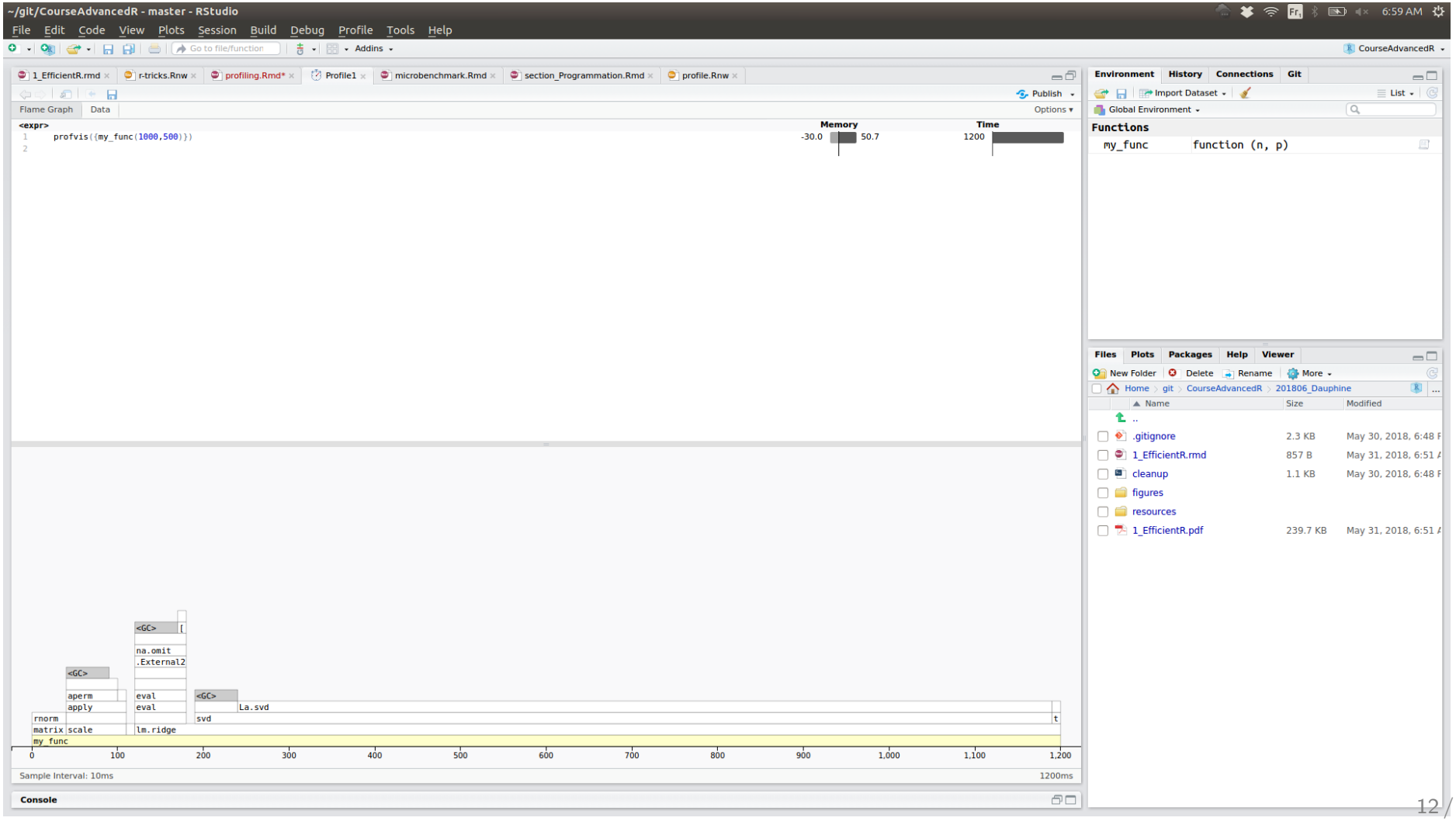
```
library(profr)
profiling <- profr({my_func(1000,500)}, interval = 0.01)
plot(profiling)
```



Profile your code within R Studiow with `profvis`

Profvis integrates the profiling to the Rstudio API

```
library(profvis)
profvis({my_func(1000,500)})
```



Outline

Outline

Outline

Outline

References

Gillespie, C., & Lovelace, R. (2016). *Efficient r programming*. " O'Reilly Media, Inc." Retrieved from https://bookdown.org/csgillespie/efficientR/

Wickham, H. (2014). *Advanced r*. CRC Press. Retrieved from http://adv-r.had.co.nz/