Lab 09

CB

2022-11-04

```
#Playing with examples from Lecture Slides
#Create the cluster
library(parallel)
cl <-makePSOCKcluster(4)
x<- 20

#Prepare the cluster
clusterSetRNGStream(cl, 123)
 #same as "set.seed(123)"
clusterExport(cl, "x")
clusterEvalQ(cl, {paste0("Hello from process #", Sys.getpid(), ". I see x and it is equal to  ", x)})
```

```
## [[1]]
## [1] "Hello from process #24780. I see x and it is equal to  20"
##
## [[2]]
## [1] "Hello from process #1032. I see x and it is equal to  20"
##
## [[3]]
## [1] "Hello from process #11716. I see x and it is equal to  20"
##
## [[4]]
## [1] "Hello from process #13728. I see x and it is equal to  20"
```

```
#Stop cluster
stopCluster(cl)
```

Problem 2

```
set.seed(1235)
fun1 <- function(n = 100, k = 4, lambda = 4) {
  x <- NULL

  for (i in 1:n)
    x <- rbind(x, rpois(k, lambda))

  return(x)
}
f1 <- fun1(100,4)
mean(f1)
```

```
## [1] 4.075
```

```
fun1alt <- function(n = 100, k = 4, lambda = 4) {
  # YOUR CODE HERE

  x <- matrix( rpois(n*k, lambda) , ncol = 4)

  return(x)
}
f1 <- fun1alt(50000,4)


# Benchmarking
microbenchmark::microbenchmark(
  fun1(),
  fun1alt()
)
```

```
## Unit: microseconds
##       expr   min    lq    mean median    uq    max neval
##     fun1() 186.0 203.9 245.029  240.7 268.6  464.5   100
##  fun1alt()  14.6  15.3  35.009   15.6  16.6 1806.6   100
```

```
 d <- matrix(1:16, ncol=4)
print(d)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

###Problem 2, find max

```
set.seed(1234)
M <- matrix(runif(12), ncol=4)
M
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.4729098 0.4138440 0.5147738 0.6423320
## [2,] 0.7697837 0.5610036 0.4077538 0.4666187
## [3,] 0.2160154 0.9327223 0.8228749 0.9267893
```

```
#Find each column max value, the 2 is the margin argument which tells R to apply that function to each
fun2 <- function(x) {
  apply(x, 2, max)}
fun2(x=M)
```

```
## [1] 0.7697837 0.9327223 0.8228749 0.9267893
```

```
fun2alt <- function(x) {
  # YOUR CODE HERE
   idx <- max.col( t(x))
   x[cbind(idx,1:4)]
}
fun2alt(x=M)
```

```
## [1] 0.7697837 0.9327223 0.8228749 0.9267893
```

```
x <- matrix(rnorm(1e4), nrow=10)
```

```
# Benchmarking
microbenchmark::microbenchmark(
  fun2(x),
  fun2alt(x)
)
```

```
## Unit: microseconds
##        expr   min     lq    mean median     uq    max neval
##     fun2(x) 562.5 583.80 766.983  662.1 834.30 2562.4   100
##  fun2alt(x)  60.9  77.45 113.944   87.2 112.95 1891.7   100
```

Problem 3, Parallelize everything

```
my_boot <- function(d, stat, R, ncpus = 1L) {

  # Getting the random indices
  n <- nrow(dat)
  idx <- matrix(sample.int(n, n*R, TRUE), nrow=n, ncol=R)
  #Step 1
  cl <-makePSOCKcluster(4)
  clusterSetRNGStream(cl, 123)
  #Step 2
  clusterExport(cl, c("stat", "dat", "idx"), envir=environment())
  #Step 3 replaces with parLapply

   ans <- parLapply(cl, seq_len(R), function(i) {
    stat(dat[idx[,i], , drop=FALSE])
  })

  # Coercing the list into a matrix
  ans <- do.call(rbind, ans)
  ans}
```

1. Use the previous pseudocode, and make it work with parallel. Here is just an example for you to try.

```
my_stat <- function(d) coef(lm(y~x, data=d))
set.seed(1)
n<-500; R<-1e4

x<-cbind(rnorm(n)); y<-x*5 + rnorm(n)

ans0<- confint(lm(y~x))
set.seed(1)
dat <-data.frame(x,y)
ans1<- my_boot(dat, my_stat, R=R, ncpus = 2L)
```

```
t(apply(ans0, 2, quantile, c(.025,.975)))
```

```
##             2.5%    97.5%
## 2.5 %  0.1252641 4.790269
## 97.5 % 0.2956810 4.963477
```

```
t(apply(ans1, 2, quantile, c(.025,.975)))
```

```
##                    2.5%     97.5%
## (Intercept) 0.003821335 0.1720413
## x           4.909348399 5.0899785
```

Check whether your version actually goes faster than the non-parallel version:

```
system.time(my_boot(dat, my_stat, R = 4000, ncpus = 1L))
```

```
##    user  system elapsed
##    0.15    0.00    0.88
```

```
system.time(my_boot(dat, my_stat, R = 4000, ncpus = 2L))
```

```
##    user  system elapsed
##    0.06    0.02    2.45
```