HW 6

Yoonseo Mok

https://github.com/mokys1213/STATS-506-FA-2024/blob/main/HW6/hw6.pdf

```
library(DBI)
library(parallel)
library(future)
library(data.table)
```

```
# Import the SQLite database of the Lahman data
lahman <- dbConnect(RSQLite::SQLite(), "lahman_1871-2022.sqlite")

# Reading in fielding data
Fielding=dbGetQuery(lahman, "SELECT * FROM Fielding")
# Calculating Range Factor
Fielding$RF=3*(Fielding$PO+Fielding$A)/Fielding$InnOuts
# Data cleaning
fielding=na.omit(Fielding[, c("teamID", "RF")])
fielding=fielding[!is.infinite(fielding$RF),]

# Calculating the average RF for each team
point_est=aggregate(fielding$RF,by=list(fielding$teamID), FUN =mean)
colnames(point_est)=c("teamID", "meanRF")
```

A

```
fielding_dat=data.table(fielding)

# Bootstrap using data.table
f = function(data) {
  tmp=data[,.SD[sample(.N, replace=TRUE)], by=teamID]
  return(tmp[,.(meanRF = mean(RF)), by=teamID])
}

# Checking the function
f(fielding_dat)
```

```
##      teamID    meanRF
##      <char>     <num>
##   1:    SFN 0.4068948
##   2:    CHN 0.4077714
##   3:    CHA 0.4032719
```

```
##   4:    BOS 0.3984979
##   5:    SEA 0.3782366
##  ---
## 136:    NYP 0.4314161
## 137:    PHP 0.4875463
## 138:    IN1 0.3858836
## 139:    RIC 0.6683865
## 140:    BL4 0.4801627
```

```
# Measuring time
system.time(f(fielding_dat))
```

```
##    user  system elapsed
##   0.012   0.000   0.012
```

A1: Without any parallel processing

```
reps=1000

# Using lapply
system.time({
  result1=lapply(seq_len(reps), function(x) f(fielding_dat))
})
```

```
##    user  system elapsed
##  10.977   0.254  11.231
```

A2: Using parallel processing with the parallel package

```
# Using mcapply
system.time({
  result2=mclapply(seq_len(reps), function(x) f(fielding_dat), mc.cores=5)
})
```

```
##    user  system elapsed
##   7.669   0.375   2.704
```

A3: Using futures with the future package

```
# Using future
plan(multisession)
system.time({
  result3=lapply(seq_len(reps), function(x) {
    future(f(fielding_dat),seed=TRUE)
  })
  result3 =lapply(result3,value)
})
```

```
##    user  system elapsed
##  56.192   1.567  72.370
```

B

```
# Generating a table showing the estimated RF and associated SE from the three approaches.
tablesum <- function(results) {
  bind=rbindlist(results)
  summary=bind[, .(meanRF=mean(meanRF), SE=sd(meanRF)), by=teamID]
  tabres =summary[order(-meanRF)][1:10] # only the top 10 teams
  return(tabres)
}

# Showing the results
tablesum(result1)
```

```
##     teamID    meanRF         SE
##     <char>     <num>      <num>
##  1:    RC1 0.5745033 0.08121311
##  2:    LS1 0.5309112 0.05809380
##  3:    MLU 0.5218507 0.12778485
##  4:    ELI 0.5213624 0.06287898
##  5:    RIC 0.5092577 0.06932807
##  6:    KEO 0.5083398 0.11272136
##  7:    BLA 0.4943952 0.03216539
##  8:    LS3 0.4893946 0.01541042
##  9:    TRN 0.4817399 0.02967563
## 10:    PHU 0.4801040 0.04452198
```

```
tablesum(result2)
```

```
##     teamID    meanRF         SE
##     <char>     <num>      <num>
##  1:    RC1 0.5751929 0.07955004
##  2:    LS1 0.5318642 0.05531692
##  3:    ELI 0.5267890 0.06427128
##  4:    MLU 0.5111600 0.12638959
##  5:    KEO 0.5111001 0.11384072
##  6:    RIC 0.5051792 0.06592793
##  7:    BLA 0.4971186 0.03180024
##  8:    LS3 0.4890533 0.01516937
##  9:    TRN 0.4814851 0.02889911
## 10:    PHU 0.4783717 0.04849758
```

```
tablesum(result3)
```

```
##     teamID    meanRF         SE
##     <char>     <num>      <num>
##  1:    RC1 0.5742484 0.08144747
##  2:    ELI 0.5309484 0.06335519
```

```
##  3:    LS1 0.5296941 0.05681483
##  4:    RIC 0.5117349 0.06753814
##  5:    KEO 0.5080258 0.11314094
##  6:    MLU 0.5035203 0.12678229
##  7:    BLA 0.4936408 0.03281496
##  8:    LS3 0.4900619 0.01565010
##  9:    TRN 0.4810074 0.02890364
## 10:    PHU 0.4804058 0.04711168
```

C

Without Parallel Processing (lapply) it took 11.7 seconds for 1000 simulations
Using parallel (mclapply) it took 2.6 seconds for 1000 simulations
Using future Package, it took 72.8 seconds for 1000 simulations

Using parallel (mclapply) is the best choice for this problem. It is the fastest.

But all three methods resulted similar mean estimates and standard errors