Results_Resample

Due May 3, 2019

Contents

Data Generation Functions 1
Seeding Functions 1
Simulation Functions 1
Bayesian Functions 1
Resampling Functions 1
MLB Data 1
Results 1
Resample_results 1
Some NLB 4

Data Generation Functions

Seeding Functions

Simulation Functions

Bayesian Functions

Resampling Functions

MLB Data

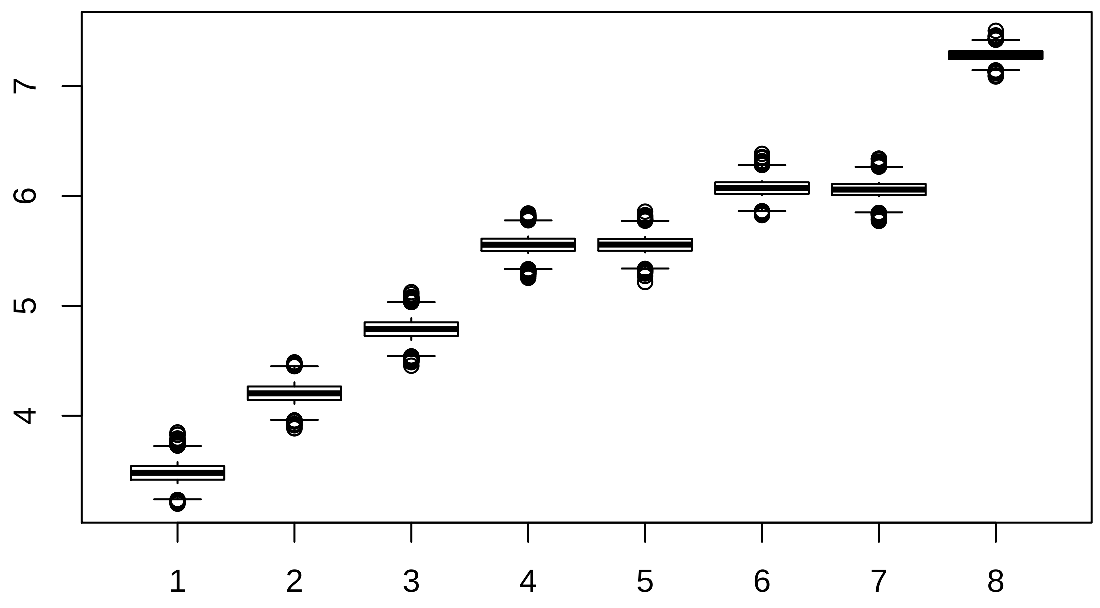
Results

Resample_results

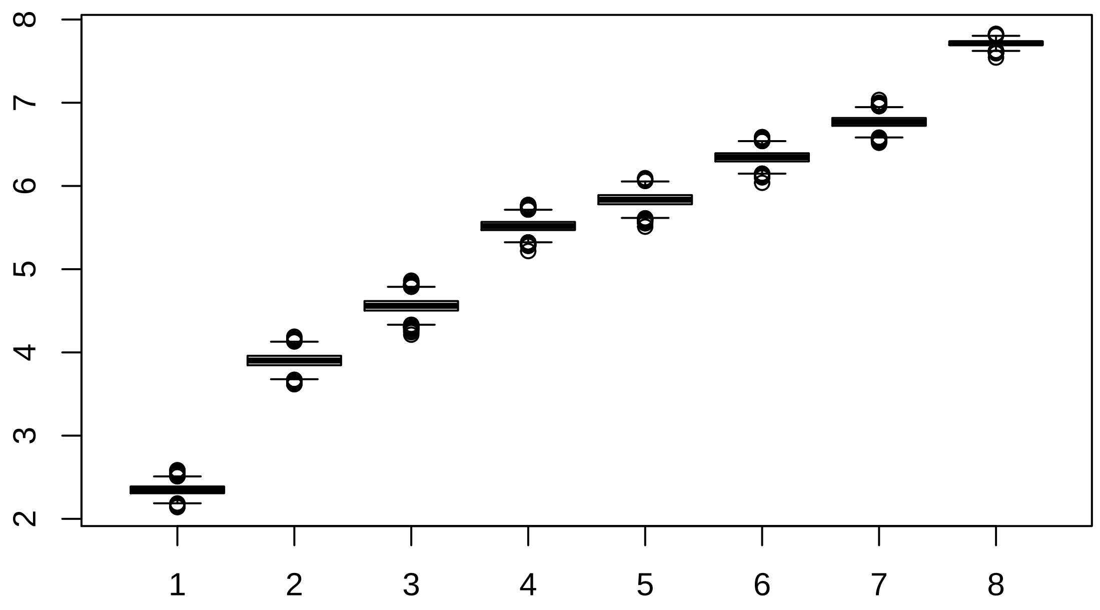
```
#another version of simTournament
#returns the final standing of teams
#for example: 2 3 1 4
#means team2 is winner, team3 is second,...
simTournament <- function(bracket, matchups) {
  # function to simulate a tournament given a bracket and probability matrix

  # INPUT:
  # bracket is the bracket structure
  # matchups is the matchup/probability matrix

  # OUTPUT:
  # final standing of teams

  nTeams <- length(bracket)
  newBracket <- bracket
  nRounds <- log2(nTeams)
  #results = c("Winner"=-1,"WinsPerTeam"=list(rep(0, ncol(matchups))))
  results <- numeric(nTeams)

  for (round in 1:nRounds) {
    #winsPrev = results["WinsPerTeam"]
    oldBracket <- newBracket
    newBracket <- simRound(newBracket, matchups)
    results[(2^(nRounds-round)+1):2^(nRounds-round+1)] <-
      setdiff(oldBracket,newBracket)
  }

  results[1] <- newBracket
  return(results)
}



#INPUT: x is index of the team
#       res is simulation results
#OUTPUT: rank of x in simulations
getRank <- function(x, res, lower = T){
  n <- length(res)
  rank <- numeric(n)
  for(i in 1:n){
    roughrank <- which(res[[i]]==x)
    if(roughrank == 2 | roughrank == 1)
      rank[i] <- roughrank
    #pick lowest rank for a range of rank
    #for example, if rank is 5th-8th, let rank be 8
    else if(lower == T)
      rank[i] <- 2^ceiling(log2(roughrank))
    #if lower = F pick highest rank in range
    else
      rank[i] <- 2^floor(log2(roughrank))+1
  }
  rank
}
```

```
numTeams = 8
nSim = 1000

#set.seed(428)
power <- genNormalPowers(numTeams)
prob = genCrossTeamWinningProbabilities(power)
#with random seeding
res = replicate(nSim, simTournament(
  sample(1:numTeams), prob),simplify = FALSE)
#ranks of each team
ranks <- lapply(1:numTeams, getRank, res = res)
#bootstrap estimation on mean of ranks of each team
mean_boot <- lapply(ranks, boot_replicate)
#a box plot of ranking of each team
mean_boot_raw <- lapply(mean_boot, function(x) x$boot)
```

```
boxplot(mean_boot_raw)
```



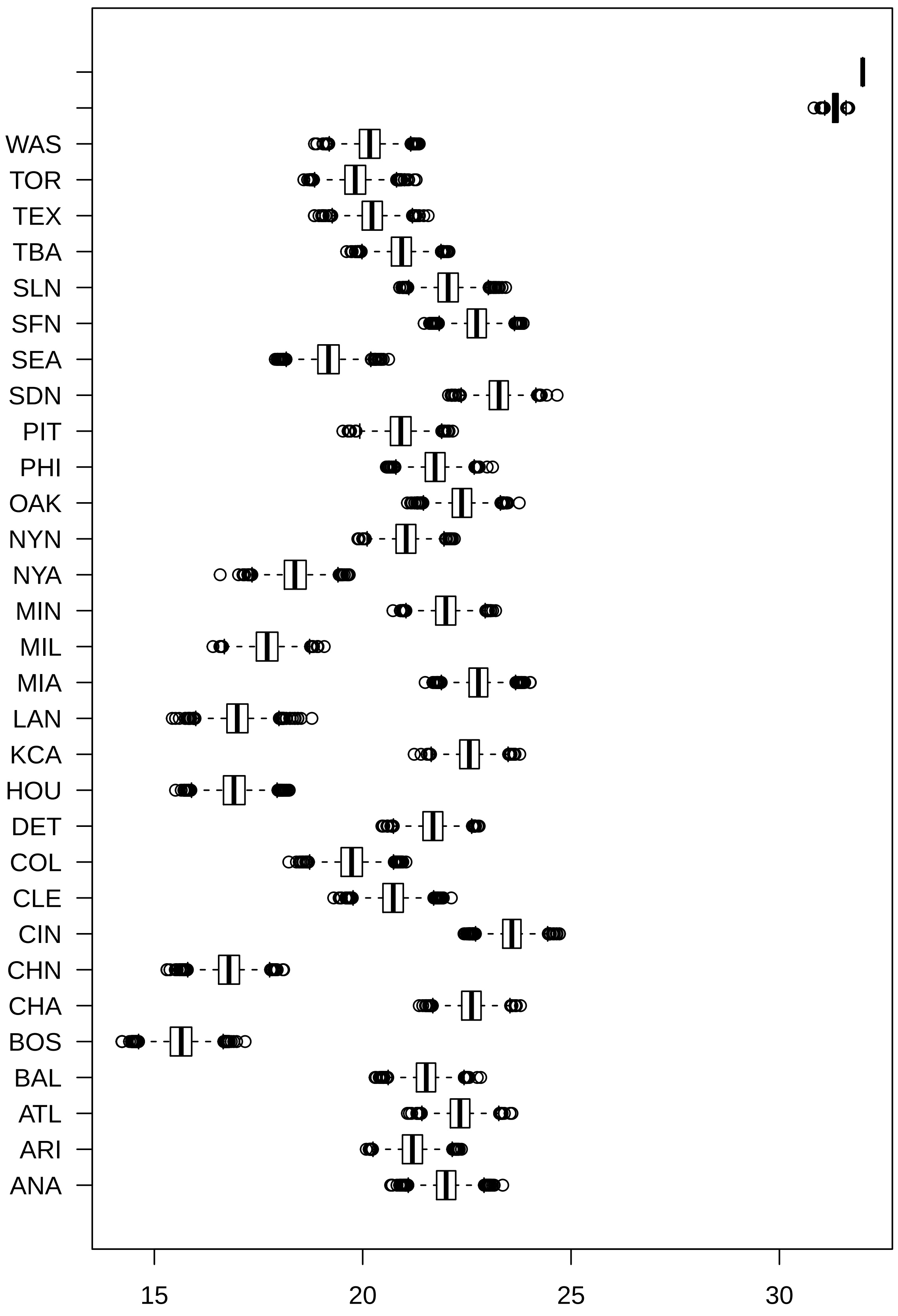
```
#with seeding
seeds = tournament_seeding(numTeams)
res = replicate(nSim, simTournament(seeds, prob), simplify = FALSE)
#ranks of each team
ranks <- lapply(1:numTeams, getRank, res = res)
#bootstrap estimation on mean of ranks of each team
mean_boot <- lapply(ranks, boot_replicate)
#a box plot of ranking of each team
mean_boot_raw <- lapply(mean_boot, function(x) x$boot)
boxplot(mean_boot_raw)
```



Some NLB

```r
nSim = 1000
#add 2 dummy team as there're 30 teams in NLB
numTeams <- 32
prob <- cbind(rbind(winprob,numeric(30),numeric(30)),rep(1.,32),rep(1.,32))
prob[32,31] <- 0

res = replicate(nSim, simTournament(
  sample(1:numTeams), prob),simplify = FALSE)
#ranks of each team
ranks <- lapply(1:numTeams, getRank, res = res)
#bootstrap estimation on mean of ranks of each team
mean_boot <- lapply(ranks, boot_replicate)
#a box plot of ranking of each team
mean_boot_raw <- lapply(mean_boot, function(x) x$boot)
names(mean_boot_raw) <- dimnames(prob)[[1]]
```

```r
boxplot(mean_boot_raw,horizontal = T, las = 1)
```


WAS
TOR
TEX
TBA
SLN
SFN
SEA
SDN
PIT
PHI
OAK
NYN
NYA
MIN
MIL
MIA
LAN
KCA
HOU
DET
COL
CLE
CIN
CHN
CHA
BOS
BAL
ATL
ARI
ANA
15
20
25
30