Working with collaborators (including through Overleaf)

August 3, 2022

You may work with a group of co-authors and share your project's folder somehow. It may be with GitHub, Dropbox, or a similar file sharing system. It is possible that the work is divided in such a way that some collaborators work only on the text, and are not involved in, or just cannot run the code. For example, they may not have Python, R, Julia, or Matlab on their computer.

It is possible for them to change into cache mode every time they compile the document, but it is cumbersome, and since the files are shared it means that they change the setting for all other collaborators. In this document we describe how such a group can work efficiently so that the coders can compile in server-mode, and the non-coders will always compile in cache mode.

We use a simulation to find the answer. First, we define a function to simulate a game:

```
using Random; Random.seed!(1)
function whichDoor( ; nds=3)
      = fill(" ", nds)
      = rand(1:nds)
     [ ] = " "
    if  [ ] == " "
        host = rand(setdiff(1:nds,  ))
    else
        host = rand(setdiff(1:nds, [ ,  ]))
    end
      = rand(setdiff(1:nds, [ , host]))
    return ( = ,  = ,  = , host=host)
end
```

```
# look at ten games
for i in 1:10
    println(whichDoor(rand(1:3)))
end
```

```
set.seed(1)
whichDoor = function(choice, nds=3) {
    doors = rep("goat", nds)
    car = sample(1:nds, 1)
    doors[car] = "car"
    if (doors[choice] == "car") {
        host = sample((1:nds)[-choice], 1)
    } else if (nds == 3){
        host = (1:nds)[-c(choice, car)]
    } else {
        host = sample((1:nds)[-c(choice, car)], 1)
    }
    if (nds ==3) {
        switch = (1:nds)[-c(choice, host)]
    } else {
    switch = sample((1:nds)[-c(choice, host)], 1)
    }
    return(c(choice=choice, car=car, switch=switch, host=host))
}

# look at ten games
for (i in 1:10)
    print(whichDoor(sample(1:3, 1)))
```

Here are the results for the ten games: montyhall-J1 montyhall-R1

Now let's define a function to count the frequency from a larger number of the simulated games.

```
function countMTH(n; nds=3)
    n_keep, n_switch = 0, 0
    for i in 1:n
        game = whichDoor(rand(1:3), nds=nds)
        if game.  == game.
            n_keep += 1
        elseif game.  == game.
            n_switch += 1
        end
    end
    return (n_keep, n_switch) ./ n
end

# simulate 100 games to approximate the probabilities
probabilities = countMTH(100)
```

```
countMTH = function(n, nds=3){
    games = replicate(n, whichDoor(sample(1:3, 1), nds=nds))
    mean(games[1,] == games[2,])
    mean(games[3,] == games[2,])
    return (c(mean(games[1,] == games[2,]), mean(games[3,] == games[2,])))
    }

# simulate 100 games to approximate the probabilities
probabilities = paste(countMTH(100), collapse=", ")
```

The approximate probabilities of keeping the original choice and switching are Error in cat(probabilities) : object 'probabilities' not found (0.32, 0.68).