Snakes and Ladders

Suraj Nath

Feb. 19, 2017

Snakes and Ladders Game Simulation with R

Introduction

Snakes and Ladders is a very simple board game between two or more players. The game has numbered squares that the player moves through by rolling a 6 sided dice but if a player lands on a certain square they get moved up (ladder) or down (snakes). This continues on until the a player reaches the last numbered square.

The ladders and slides/snakes will be set up as a data frame,

```
ladder.df <- data.frame(start=c(1,4,9,21,28,36,51,71,80), end=c(38,14,31,42,84,44,67,91,100))
slide.df <- data.frame(start=c(98,95,93,87,64,62,56,49,47,16), end=c(78,75,73,24,60,19,53,11,26,6))
```

```
library(knitr)
```

The ladders:

```
kable(ladder.df, align="c")
```

start	end
1	38
4	14
9	31
21	42
28	84
36	44
51	67
71	91
80	100

The Snakes:

```
kable(slide.df, align="c")
```

start	end
98	78
95	75
93	73
87	24
64	60
62	19
56	53
49	11
47	26

start	end
16	6

```
curLoc <- 0 # Current location
nroll <- 0 # Number of rolls
slides <- 0 # Number of slides encountered
ladders <- 0 # Number of ladders encountered
# Keep rolling dice and moving until reach 100 or greater ending the game
while(curLoc < 100) {
  roll <- sample(6,1) # generate random number between [1 to 6]
  curLoc <- curLoc + roll # increase position
  nroll <- nroll + 1 # increase number of rolls
  # Need to check if we landed on a ladder or slide and move forward or back
  if (any(ladder.df$s %in% curLoc)) {
    curLoc <- ladder.df$e[ladder.df$s %in% curLoc]
    ladders <- ladders + 1
  }
  if (any(slide.df$s %in% curLoc)) {
    curLoc <- slide.df$e[slide.df$s %in% curLoc]
    slides <- slides + 1
  }
}
```

```
library(foreach)
library(doParallel)
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
registerDoParallel(cores=4)
```

```
getDoParWorkers()
```

```
## [1] 4
```

```
num.iter <- 500 # Number of play throughs

# Get timing as well
stime <- system.time({
  out.seq <- foreach(icount(num.iter), .combine=rbind) %do% {
    curLoc <- 0
    nroll <- 0
    slides <- 0
    ladders <- 0
    # Keep rolling dice and moving until reach 100 or greater ending the game
    while(curLoc < 100) {
      roll <- sample(6,1) # generate random number between [1 to 6]
      curLoc <- curLoc + roll # increase position
      nroll <- nroll + 1 # increase number of rolls
      # Need to check if we landed on a ladder or slide and move forward or back
      if (any(ladder.df$s %in% curLoc)) {
        curLoc <- ladder.df$e[ladder.df$s %in% curLoc]
        ladders <- ladders + 1
      }
```

```
   if (any(slide.df$s %in% curLoc)) {
    curLoc <- slide.df$e[slide.df$s %in% curLoc]
    slides <- slides + 1
   }
  }
  # Create output to store, num rolls, num ladders hit, num slides hit
  out.info <- c(nroll, ladders, slides)
 }})[3]
```

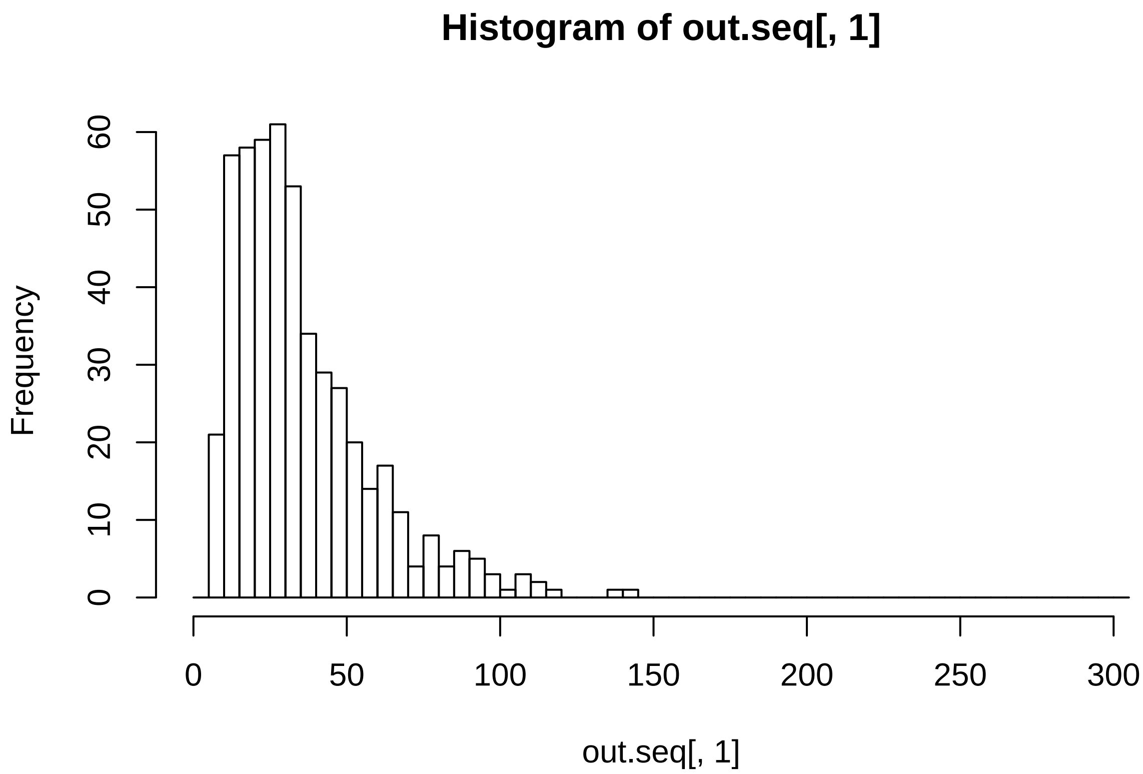
Time taken by simulation

```
stime
```

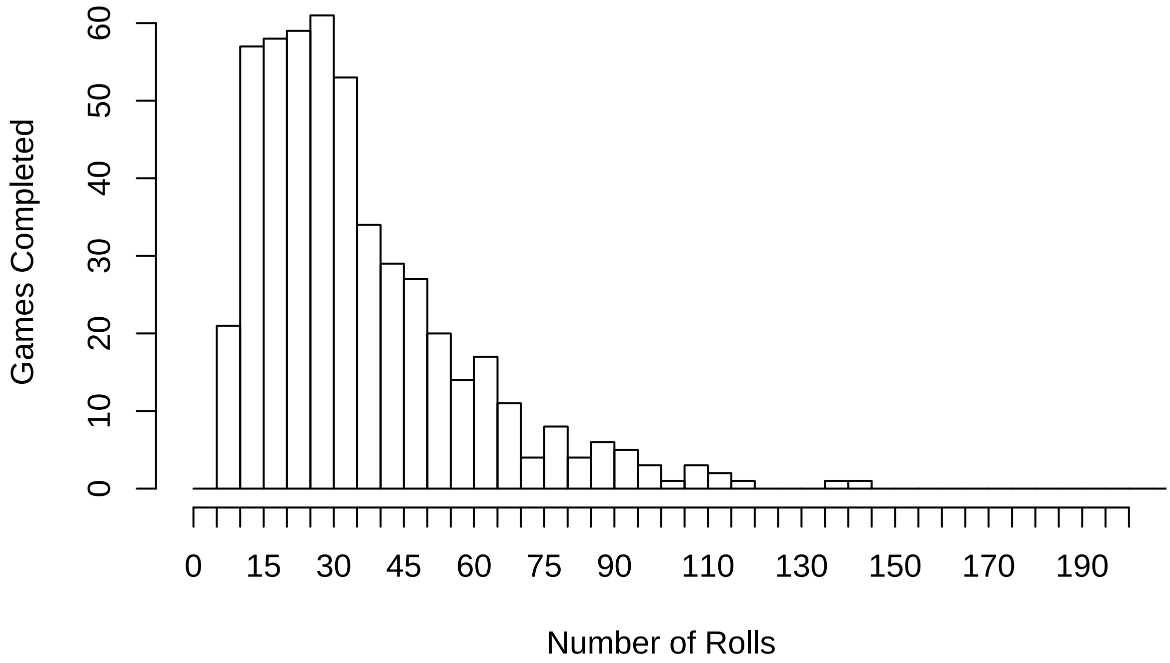
```
## elapsed
##    7.95
```

Take a look at a histogram of number of rolls to complete the game

```
h <- hist(out.seq[,1], breaks=seq(0,305,5))
```

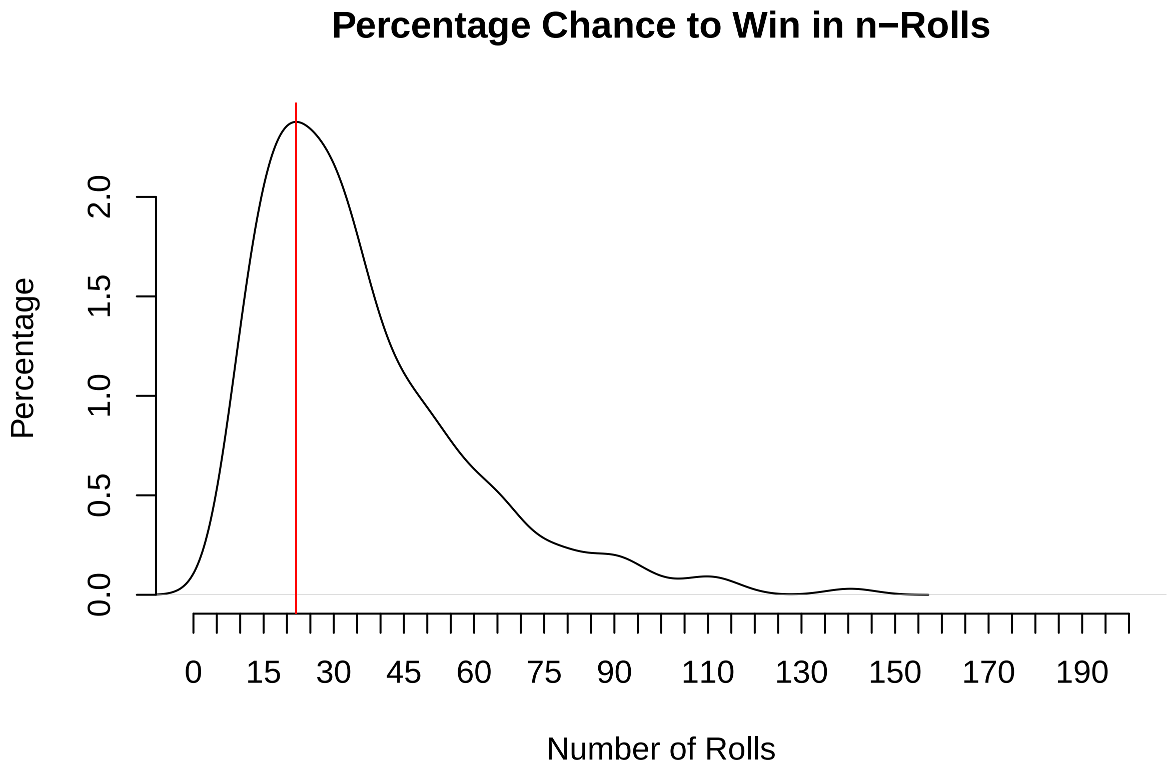


```
plot(h, main="", xlab="Number of Rolls", ylab="Games Completed", xlim=c(0,200), axes=F)
axis(1, at=seq(0,200,5))
axis(2)
```



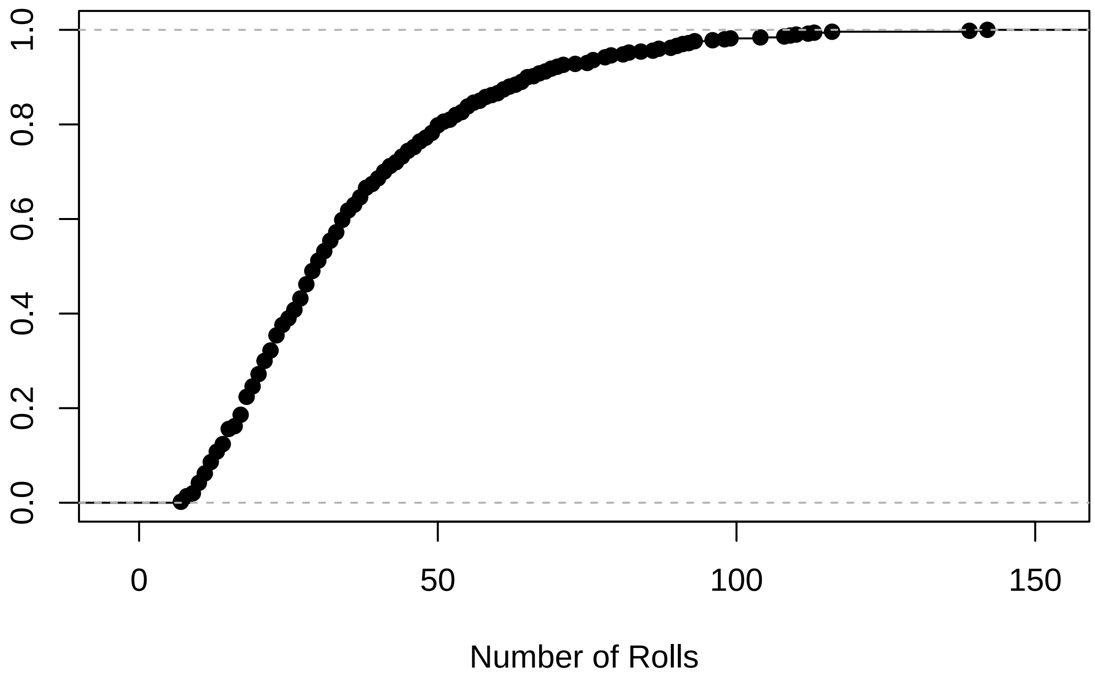
Plot for percentage chance to win the game in n rolls:

```
d <- density(out.seq[,1])
d$y <- d$y * 100
plot(d, main="Percentage Chance to Win in n-Rolls", xlab="Number of Rolls", ylab="Percentage", xlim=c(0
axis(2)
axis(1, at=seq(0,200,5))
abline(v=d$x[which.max(d$y)], col="red")
```



cumulative distribution function:

```
plot(ecdf(out.seq[,1]), xlab="Number of Rolls", ylab="",main="")
```


0.0
0.2
0.4
0.6
0.8
1.0
0
50
100
150
Number of Rolls