Knight Path Expectation

James

2/8/2022

Summary

I use R to simulate a knight moving randomly on a chessboard to see the average (mean) time it takes for a knight to visit each space. THe simulation predicts a mean of about 540 moves with a standard deviation of around 180 indicating that the sample mean occurs between 180 and 900 about 97% of the time. This format is a markdown file, so the code will run again as I publish it into pdf which will result in some small changes in the simulation.

Plan

I simulate the board with an 8x8 matrix so A1 is matrix position 1,1. I did not feel like programming the if else tree for all legal knights moves. Instead my simulated knight picks one of its 8 moves at random, and then that candidate move is validated using acceptance/rejeciton criteria of ‘being on the board’. If it is valid the knight takes that move, it is not the knight picks again. This is methodologically the same as acceptance/rejection sampling in bootstrap.

Code

```
rando_knight <- function(iterations=10){

  iterations <- iterations # the numer of iterations you want to see

  path_length <- rep(0,iterations) # initialize the vector that will hold our path lengths for e
ach vector


for (iter in 1:iterations) {

  # Set Initial Postion for Knight and clear the board
  state_board = matrix(0,nrow = 8, ncol = 8)
  state_board[1,1] <- 1
  row_pos = 1
  col_pos = 1

  # Set Completion Status
  complete_state <- 0 # will turn to 1 once knight touches every square
  path_length_counter <- 0

  # Move Around until Knight Goes to Every Spot
  while (complete_state == 0) {

    move_succes <- 0

    while (move_succes==0) {

      ### Move the Knight Quick to Code but needs validation (see below)
      x<-sample(1:8,1) # draw a random number between 1 and 8
      ### Forcast the knight's acording to random number.

      if (x==1) {
        temp_row <- row_pos+2
        temp_col <- col_pos+1
      }

      if (x==2) {
        temp_row <- row_pos+1
        temp_col <- col_pos+2
      }

      if (x==3) {
        temp_row <- row_pos-2
        temp_col <- col_pos+1
      }

      if (x==4) {
        temp_row <- row_pos-1
        temp_col <- col_pos+2
      }

      if (x==5) {
```

```
        temp_row <- row_pos+2
        temp_col <- col_pos-1
      }

      if (x==6) {
        temp_row <- row_pos+1
        temp_col <- col_pos-2
      }

      if (x==7) {
        temp_row = row_pos-2
        temp_col = col_pos-1
      }

      if (x==8) {
        temp_row = row_pos-1
        temp_col = col_pos-2
      }

      ### Validate that the move is legal aka on the boar.
      if ((temp_row <=8)&(temp_row >=1)&(temp_col <=8)&(temp_col >=1)) {
        row_pos <- temp_row
        col_pos <- temp_col
        move_succes <- 1
        path_length_counter <- path_length_counter+1
      }

    }

    ### Update Stateboard
    state_board[row_pos,col_pos] <- 1

    ### Check Completion COnditions
    if (sum(state_board)==64) {

      complete_state <- 1

    }



  }

  path_length[iter]<-path_length_counter

}
return(path_length)

}
```

Results

Now I run the experiment by seeing how long it takes 1,000 knights to visit every spot on the board.

```
set.seed(2022) # set a seed for reproducibility.
path_lengths <- rando_knight(1000)
```

Here I publish the summary statistics

```
summary(path_lengths)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   226.0   413.0   501.0   545.0   621.5  1858.0
```

```
print('standard Deviation:')
```

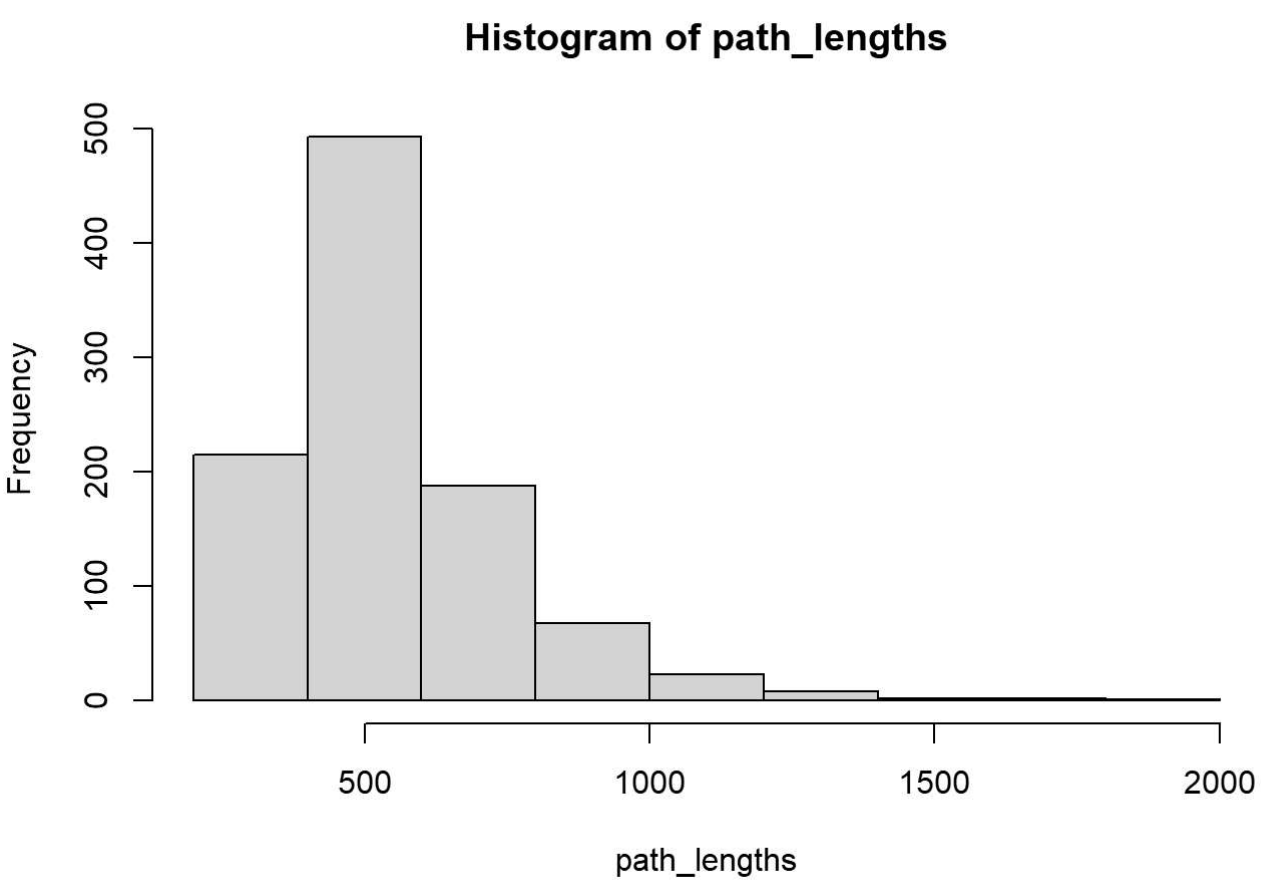
```
## [1] "standard Deviation:"
```

```
sqrt(var(path_lengths))
```

```
## [1] 199.5566
```

Here is histogram of the results.

```
hist(path_lengths)
```


Histogram of path_lengths
Frequency
0
100
200
300
400
500
500
1000
1500
2000
path_lengths