Running a Simulation Study - Solutions

Exercise A - (3 min)

Recall this code snippet from our earlier lecture on simulation:

```
dice_sum <- \() {
# Roll a pair of fair, six-sided dice and return their sum
  die1 <- sample(1:6, 1)
  die2 <- sample(1:6, 1)
  die1 + die2
}
sims <- map_dbl(1:10000, \(i) dice_sum())
```

1. Use your new-found knowledge of `purrr` to explain line 7.
2. Write a `for` loop to replace line 7.

Solution

The anonymous function `\(i) dice_sum()` has one argument: `i`. But this argument isn't used in any way! Regardless of the value of `i` we simply call `dice_sum()`. This is just a sneaky way of getting `map_dbl()` to *repeatedly* call `dice_sum()` a total of `10000` times. It is equivalent to the following `for()` loop:

```
nreps <- 10000
sims <- rep(NA_real_, nreps)
for(i in seq_along(sims)) {
  sims[i] <- dice_sum()
}
```

Exercise B - (35 min)

1. Write a function called `get_avg_after_streak()`. Given a vector `shots`, it should construct `after_streak` with $k = 3$ and return `mean(after_streak)`. Test your function using `c(0, 1, 1, 1, 1, 0, 0, 0)`.
2. Write a function called `draw_shots()` that simulates a sequence of `n_shots` iid Bernoulli trials, each with probability of success `prob_success`.
3. Amos knows that Liz makes 1/2 of her shots on average. He watches her take 100 shots and then computes $T =$ `get_avg_after_streak()`, where a streak is defined by $k = 3$. Amos argues: "If Liz does **not have a hot hand**, then each shot is independent of the others with probability of success 1/2. This means the expected value of T will be 1/2." Carry out a simulation with 10,000 replications to check Amos' argument.
4. Repeat the preceding over a parameter grid with `n_shots` $\in \{50, 100, 200\}$ and `prob_success` $\in \{0.4, 0.5, 0.6\}$. What do you conclude from your simulation?

Solution

Part 1

```
get_avg_after_streak <- function(shots) {

  # shots should be a vector of 0 and 1; if not STOP!
  stopifnot(all(shots %in% c(0, 1)))

  n <- length(shots)
  after_streak <- rep(NA, n) # Empty vector of length n

  # The first 3 elements of shots by definition cannot
  # follow a streak
  after_streak[1:3] <- FALSE

  # Loop over the remaining elements of shots
  for(i in 4:n) {
    # Extract the 3 shots that precede shot i
    prev_three_shots <- shots[(i - 3):(i - 1)]
    # Are all three of the preceding shots equal to 1?
    # (TRUE/FALSE)
    after_streak[i] <- all(prev_three_shots == 1)
  }

  # shots[after_streak] extracts all elements of shots
  # for which after_streak is TRUE. Taking the mean of
  # these is the same as calculating the prop. of ones
  mean(shots[after_streak])
}

get_avg_after_streak(c(0, 1, 1, 1, 1, 0, 0, 0))
```

```
[1] 0.5
```

Part 2

```
draw_shots <- function(n_shots, prob_success) {
  rbinom(n_shots, 1, prob_success)
}
set.seed(420508570)
mean(draw_shots(1e4, 0.5))
```

```
[1] 0.4993
```

Part 3

The average value of T appears to be noticeably *less* than 0.5:

```
library(tidyverse)
nreps <- 1e4
sim_datasets <- map(1:nreps, \(i) draw_shots(100, 0.5))
sim_estimates <- map_dbl(sim_datasets, get_avg_after_streak)
mean_T <- mean(sim_estimates)
mean_T
```

```
[1] 0.4647162
```

To check whether this discrepancy is statistically meaningful, we can construct a worst-case 99.7% confidence interval for $\mathbb{E}(T)$ as follows. The standard error of a sample proportion is $\sqrt{p(1-p)/n}$, where p is the true proportion. But the whole problem is that we don't know p: that's why we want a standard error in the first place! A simple calculation shows, however, that the standard error is *maximized* if $p = 0.5$. Hence, $1/(2 \times \texttt{nreps})$ is the *worst case* standard error. Three times this quantity is the worst-case margin of error for a 99.7% confidence interval based on the central limit theorem:

```
mean_T + c(-1, 1) * 3 / (2 * nreps)
```

```
[1] 0.4645662 0.4648662
```

This shows that we can rule out $\mathbb{E}(T)$ with *extremely high confidence* based on the number of simulation replications that we used. Amos is **wrong**. In this simulation the shots are iid coin flips and yet we found evidence of a **cold hand**. In fact, this approach is biased *against* finding evidence of a hot hand even if it exists.

Part 4

The bias appears to shrink as the length of the sequence or the probability of success increase:

```
sim_params <- expand_grid(n_shots = c(50, 100, 200),
                          prob_success = c(0.4, 0.5, 0.6))

run_sim <- \(n_shots, prob_success, nreps = 1e4) {
  map(1:nreps, \(i) draw_shots(n_shots, prob_success)) |>
    map_dbl(get_avg_after_streak)
}

sim_results <- pmap(sim_params, run_sim)

# In very short sequences there may be no streaks, leading to an NaN
# Drop these, so we effectively condition on their being at least one
# streak in the sequence
summary_stats <- map_dbl(sim_results, mean, na.rm = TRUE)
summary_stats <- sim_params |>
  bind_cols(mean_T = summary_stats) |>
  mutate(bias = mean_T - prob_success)
summary_stats |>
  knitr::kable(digits = 2)
```

n_shots	prob_success	mean_T	bias
50	0.4	0.29	-0.11
50	0.5	0.43	-0.07
50	0.6	0.55	-0.05
100	0.4	0.33	-0.07
100	0.5	0.46	-0.04
100	0.6	0.58	-0.02
200	0.4	0.37	-0.03
200	0.5	0.48	-0.02
200	0.6	0.59	-0.01