Lab 07 Jackie Stephens

Q1: What is the sample size, n? Show the code you used for the calculation and remember to check for missing data. **N = 219L**

dat_Gentoo = subset(penguins, species != "Gentoo")

n = sum(!is.na(dat_Gentoo$bill_length_mm))

My revised answer:

dat_Gentoo = subset(penguins, species == "Gentoo")

n = sum(!is.na(dat_Gentoo$bill_length_mm)) = 123

The error in my initial code relates to my subsetting. In my initial code, I used species != "Gentoo" which created a subset that excluded Gentoo when I wanted to make a subset with only Gentoo represented. I solved this by using the == instead of the opposite != .

Q2: What is the sample standard deviation? Show the code you used for the calculation.

sd(dat_Gentoo$bill_length_mm, na.rm = TRUE) = 5.477062

My revised answer:

dat_Gentoo = subset(penguins, species == "Gentoo")

sd(dat_Gentoo$bill_length_mm, na.rm = TRUE) = 3.081857

The reason I got the wrong answer initially is because I subsetted wrong at the beginning so while the code was correct the first time around, it gave me the wrong answer. So with the corrected subsetting, I was able to produce the correct answer this time around.

Q3: What are the critical t-values? Show the R code you used for the calculation.

t_crit = abs(qt(alpha / 2, df = n - 1))

-1.97 and 1.97

My revised answer:

alpha = 0.05

t_crit = abs(qt(alpha / 2, df = n - 1))

-1.9796 and 1.9796

The common theme seems to be that I got all of these wrong because of my incorrect subsetting at the beginning which lead to the wrong sample size, which then was used as the n value in the critical t-values. Running my same code with the new correct subsetting corrected all of my answers. (Im not really sure I understand how I got the right answer the first time around though..)

Q4: What is the sample standard error? Show the R code you used for the calculation.

sse = sd(dat_Gentoo$bill_length_mm, na.rm = TRUE)/sqrt(n) = 0.370

My revised answer:

```
dat_Gentoo = subset(penguins, species == "Gentoo")

sse = sd(dat_Gentoo$bill_length_mm, na.rm = TRUE)/sqrt(n)
```

0.277

Yet again the incorrect sub setting and sample size corrupted my initial answer, so with the correct subsetting my revised answer is functioning properly.

Q5: Finally, construct the CI and show the R code you used for the calculation.

CI = 0.2660114 and 1.724898

```
GentooBL_ci = c(
  lower = mean(dat_Gentoo$bill_length_mm) - ci_radius,
  upper = mean(dat_Gentoo$bill_length_mm) + ci_radius)
```

My revised answer:

```
sse = sd(dat_Gentoo$bill_length_mm, na.rm = TRUE)/sqrt(n)

t_crit = abs(qt(alpha / 2, df = n - 1))

ci_radius = sse * t_crit

mean(na.omit(dat_Gentoo$bill_length_mm))

lower = mean(na.omit(dat_Gentoo$bill_length_mm)) - ci_radius

upper = mean(na.omit(dat_Gentoo$bill_length_mm)) + ci_radius

GentooBL_ci = c(lower - ci_radius, upper + ci_radius)

GentooBL_ci =    46.40469     48.60507
```

Again initial answer was wrong because of incorrect subsample and sample size. Once corrected and addition of the na.omit to remove the NA values present in bill length measurements, I was able to get the correct answer.

Q6: What is the CI?

41.19909, 42.65024

My revised answer: GentooBL_ci = 46.40469 48.60507

See my revised code and explanation above. I solemnly swear to always check my subsetting to ensure I start off on the right foot.

Q7: Show the r code you used to call the boot() function.

Gentoo_boot = boot(data = dat_Gentoo$bill_length_mm, statistic = boot_mean, R = 10000)

Q8: Show the r code you used to calculate the upper and lower 2.5% quantiles.

quantile(Gentoo_boot$t, c(0.025, 0.975))

Q9: Show your completed rarefaction_sampler() function.

```
rarefaction_sampler = function(input_dat, n_iterations)
{
  n_input_rows = nrow(input_dat)
  n = n_input_rows
  results_out = matrix(
    nrow = n_iterations,
    ncol = n_input_rows)
  for(i in 1:n_iterations)
  {
    for(j in 1:n)
    {
```

```
    rows_j = sample(n, size = j, replace=TRUE)
    t1 = input_dat[rows_j, ]
    t2 = apply(t1, 2, sum)
    results_out[i, j] = sum(t2 > 0)
   }
  }
 return(results_out)
}
```

Q10: What did you find most difficult about building the function?

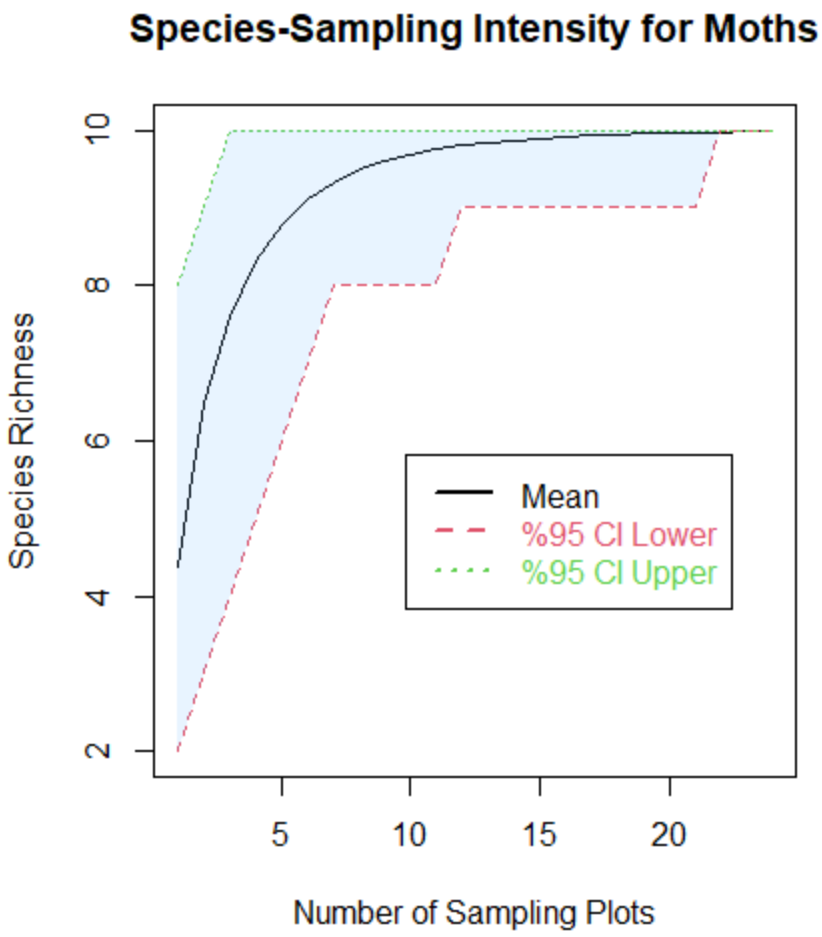
I found trying to understand how the loops worked the most difficult when building the function. Also, trying to identify what to set n to was difficult because it seemed incorrect to say it to a variable that was already defined.

Q11: Show the code you used to perform the simulations and construct the curve.

```
moths = read.csv(here("data", "moths.csv"))
rarefact = rarefaction_sampler(moths[,-1], 10000)
head(rarefact)
rare_mean = apply(rarefact, 2, mean)
rare_quant = apply(rarefact, 2, quantile, probs=c(0.025, 0.975))
rare = t(rbind(rare_mean, rare_quant))
head(rare)
matplot(
 rare,
 type='l',
 xlab='Number of Sampling Plots',
```

```
  ylab='Species Richness',
  main="Species-Sampling Intensity for Moths")
polygon(x = c(1:24,24:1),y = c(rare[,2],rev(rare[,3])),
        col= adjustcolor("dodgerblue", alpha.f = 0.10),
                    border= NA)
legend(
  'bottomright',
  legend=c('Mean','%95 CI Lower','%95 CI Upper'),
  lty=c(1,2,3),lwd=c(2,2,2), col=c(1,2,3), inset=c(0.1,0.25),
  text.col = c(1,2,3))
```

Q12: Include your rarefaction curve plot in your report. Show the R-code you used to create your plot.



Q13: About how many sites should you visit if you want to see all of the moth species? Explain your reasoning using your rarefaction curve figure.

If you wanted to see all the moth species, you would most likely want to visit at least 15 sites if not all 20. I say this because the curve shows that the confidence intervals don't start to narrow until you visit more sites. Even though the 95% CIs do not guarantee that you have a 95% chance of seeing all 10 at more sampling plots, the proximity to the mean is convincing.