Lab 08: Proportions and Inference

Solutions

- Due date: Friday, November 6 at 10:00pm.
- Submission process: Follow the submission instructions on the final page. Make sure you do not remove any `\newpage` tags or rename this file, as this will break the submission.

Helpful hints:

- Every function you need to use was taught during lecture! So you may need to revisit the lecture code to help you along by opening the relevant files on Datahub. Alternatively, you may wish to view the code in the condensed PDFs posted on the course website. Good luck!
- Knit your file early and often to minimize knitting errors! If you copy and paste code for the slides, you are bound to get an error that is hard to diagnose. Typing out the code is the way to smooth knitting! We recommend knitting your file each time after you write a few sentences/add a new code chunk, so you can detect the source of knitting errors more easily. This will save you and the GSIs from frustration! **You must knit correctly before submitting.**
- If your code runs off the page of the knitted PDF then you will LOSE POINTS! To avoid this, have a look at your knitted PDF and ensure all the code fits in the file (you can easily view it on Gradescope via the provided link after submitting). If it doesn't look right, go back to your .Rmd file and add spaces (new lines) using the return or enter key so that the code runs onto the next line.

Instructions

Tests of changes in sex ratios based on a single sample

There is a long literature studying changes in sex-ratios of births due to stressful events, such as 9/11. In today's lab, we consider a relatively small study that recorded biomarkers of stress on pregnancy. In the group of subjects that had the highest markers of stress (based on cortisol), there were 14 births to males out of a total of 38.

In this lab, we will compare the four methods we learned to calculate CIs for proportions. Recall that two of these methods involved hand calculations (though we can treat R as if it were a calculator) and two of the methods used built-in R functions.

1. Use the Normal approximation to construct a 95% confidence interval in this high stress group. We also called this specific method of constructing the CI the "large sample method". Assign the object `large_sample_ci` to a vector of the lowerbound and upperbound rounded to 4 decimals

```
###############################################

# example of final answer where 0.1234 is my
# lowerbound and 0.5678 is my upperbound
my_ci <- c(0.1234, 0.5678)

##############################################

# your code/work here

large_sample_ci <- c("YOUR CI HERE")

large_sample_ci
```

```
## [1] "YOUR CI HERE"
```

```
check_problem1()
```

```
## [1] "Checkpoint 1 Error: Did you store your CI as a numeric object?"
## [1] "Checkpoint 2 Error: Did you store 2 elements in your object?"
## [1] "Checkpoint 3 Error: Incorrect value of the interval."
##
## Problem 1
## Checkpoints Passed: 0
## Checkpoints Errored: 3
## 0% passed
## --------
## Test: FAILED
```

2. Create the 95% CI again, this time using the R function that implements the Wilson Score method with a continuity correction.

```
# your code/work here

wilson_score_CI <- c("YOUR CI HERE")
```

```r
wilson_score_CI
```

```
## [1] "YOUR CI HERE"
```

```r
check_problem2()
```

```
## [1] "Checkpoint 1 Error: Did you store your CI as a numeric object?"
## [1] "Checkpoint 2 Error: Did you store 2 elements in your object?"
## [1] "Checkpoint 3 Error: Incorrect value of the interval."
##
## Problem 2
## Checkpoints Passed: 0
## Checkpoints Errored: 3
## 0% passed
## --------
## Test: FAILED
```

3. Create the 95% CI again, this time using the Plus 4 method.

```r
# your code/work here
plus_4_ci <- c("YOUR CI HERE")


plus_4_ci
```

```
## [1] "YOUR CI HERE"
```

```r
check_problem3()
```

```
## [1] "Checkpoint 1 Error: Did you store your CI as a numeric object?"
## [1] "Checkpoint 2 Error: Did you store 2 elements in your object?"
## [1] "Checkpoint 3 Error: Incorrect value of the interval."
##
## Problem 3
## Checkpoints Passed: 0
## Checkpoints Errored: 3
## 0% passed
## --------
## Test: FAILED
```

4. Create the 95% CI again, this time using the R function that implements the Clopper Pearson (Exact) method.

```r
# your code here

exact_method_ci <- c("YOUR CI HERE")
exact_method_ci
```

```
## [1] "YOUR CI HERE"
```

```r
check_problem4()
```

```
## [1] "Checkpoint 1 Error: Did you store your CI as a numeric object?"
## [1] "Checkpoint 2 Error: Did you store 2 elements in your object?"
## [1] "Checkpoint 3 Error: Incorrect value of the interval."
##
## Problem 4
## Checkpoints Passed: 0
## Checkpoints Errored: 3
## 0% passed
## --------
## Test: FAILED
```

5. Summarize the four methods' estimates in the following table. Do they include the null hypothesized value for the sex ratio?

Method	95% Confidence Interval
Large sample	AA.AA% to AA.AA%
Wilson Score*	AA.AA% to AA.AA%
Plus four	AA.AA% to AA.AA%
Exact	AA.AA% to AA.AA%

- with continuity correction.

[TODO: Fill in the blank]

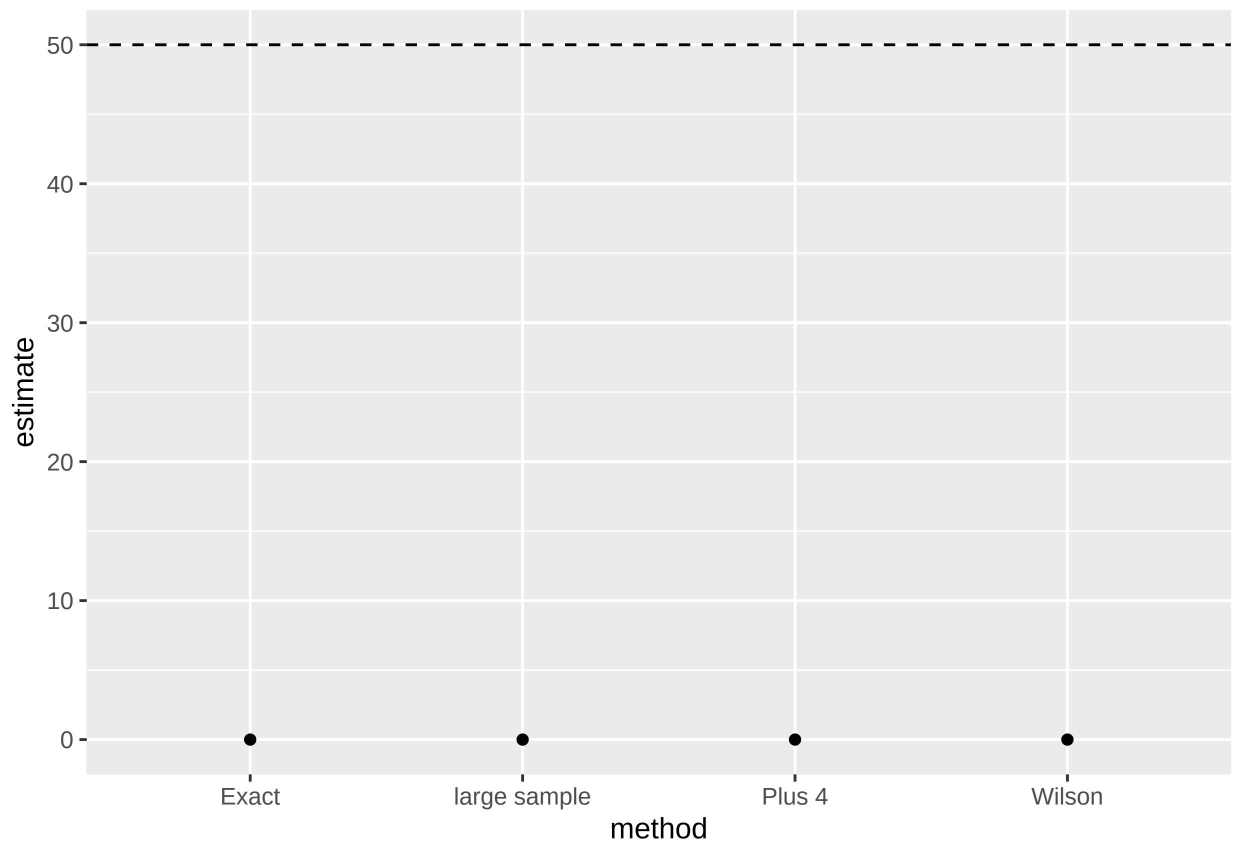
6. Here is a code template to help you to graphically present these estimates. Graphical presentations of estimates and their CIs is very useful for assessing whether the CIs overlap the null hypothesized value and tends to be better than presenting tables of estimates to readers of your research.

```r
# First make a tibble (an easy way to make a data frame) with the data about
# each confidence interval. To do this, replace each instance of 0.00 with the
# estimate from your calculations above.
library(ggplot2)
library(tibble)
```

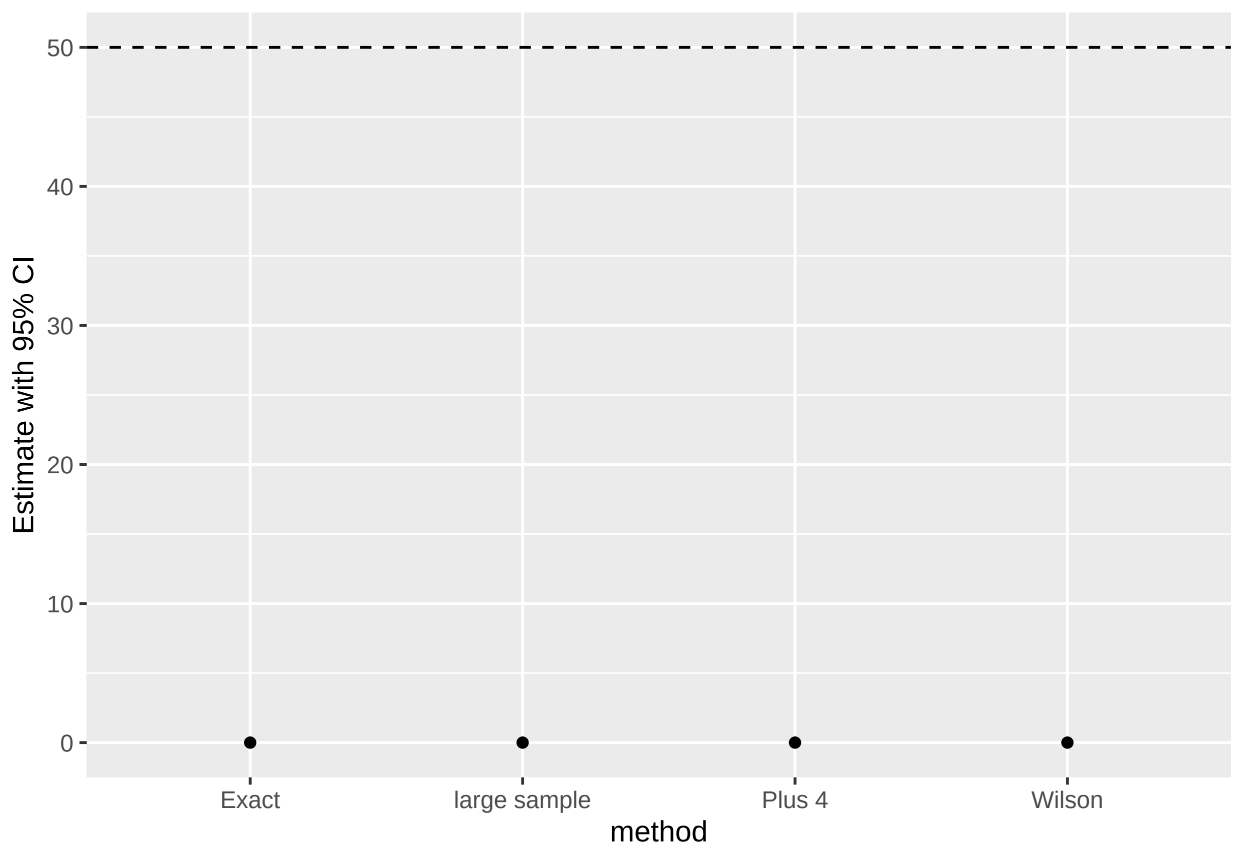
```
##
## Attaching package: 'tibble'

## The following object is masked from 'package:assertthat':
##
##     has_name
```

```r
sex_CIs <- tibble(method   = c("large sample", "Exact", "Wilson", "Plus 4"),
                  lower_CI = c(0.0           , 0.0    , 0.0     , 0.0),
                  upper_CI = c(0.0           , 0.0    , 0.0     , 0.0),
                  estimate = c(0.0           , 0.0    , 0.0     , 0.0)
                  )
# Build the ggplot incrementally, to understand how it works.
# Step 1: (qu: why do we put a horizontal line at 50?)
ggplot(data = sex_CIs, aes(x = method, y = estimate)) +
  geom_point() +
  geom_hline(aes(yintercept = 50), lty = 2)
```



```r
# Step 2:
ggplot(data = sex_CIs, aes(x = method, y = estimate)) +
  geom_point() +
  geom_hline(aes(yintercept = 50), lty = 2) +
  geom_segment(aes(x = method, xend = method, y = lower_CI, yend = upper_CI)) +
  labs(y = "Estimate with 95% CI")
```



```r
p6 <- "YOUR ANSWER HERE"


p6
```

```
## [1] "YOUR ANSWER HERE"
```

```r
check_problem6()
```

```
## [1] "Checkpoint 1 Error: You did not define a ggplot."
## [1] "Checkpoint 2 Error: Did not use breastfeed_CIs as the data"
## [1] "Checkpoint 3 Error: Did you specify the data correctly?"
## [1] "Checkpoint 4 Error: Did you specify the data correctly?"
##
## Problem 6
## Checkpoints Passed: 0
## Checkpoints Errored: 4
## 0% passed
## --------
## Test: FAILED
```

What does `geom_segment()` do? In particular, what do `x`, `xend`, `y` and `yend` specify in this case?

[YOUR ANSWER HERE]

7. Based on this plot, what can you say about the confidence intervals for the sex ratio in the high stress group? [YOUR ANSWER HERE]

8. If you have time, repeat the above analysis for the group with low stress. There were 25 births to this group, of which 17 of them were to males.

```
# your code here
```

9. If you recreated the graph for the low stress group, what can you say about the confidence intervals for the sex ratio in this group?

Check your score

Click on the middle icon on the top right of this code chunk (with the downwards gray arrow and green bar) to run all your code in order. Then, run this chunk to check your score.

```
# Just run this chunk.
total_score()
```

```
##                     Test Points_Possible          Type
## Problem 1         FAILED               1    autograded
## Problem 2         FAILED               1    autograded
## Problem 3         FAILED               1    autograded
## Problem 4         FAILED               1    autograded
## Problem 5 NOT YET GRADED               1 free-response
## Problem 6         FAILED               1    autograded
## Problem 7 NOT YET GRADED               1 free-response
## Problem 8 NOT YET GRADED               1 free-response
## Problem 9 NOT YET GRADED               1 free-response
```

Submission

For assignments in this class, you'll be submitting using the **Terminal** tab in the pane below. In order for the submission to work properly, make sure that:

1. Any image files you add that are needed to knit the file are in the `src` folder and file paths are specified accordingly.
2. You **have not changed the file name** of the assignment.
3. The file is saved (the file name in the tab should be **black**, not red with an asterisk).
4. The file knits properly.

Once you have checked these items, you can proceed to submit your assignment.

1. Click on the **Terminal** tab in the pane below.
2. Copy-paste the following line of code into the terminal and press enter.

```
cd; cd ph142-fa20/lab/lab08; python3 turn_in.py
```

3. Follow the prompts to enter your Gradescope username and password. When entering your password, you won't see anything come up on the screen–don't worry! This is just for security purposes–just keep typing and hit enter.
4. If the submission is successful, you should see "Submission successful!" appear as output.
5. If the submission fails, try to diagnose the issue using the error messages–if you have problems, post on Piazza.

The late policy will be strictly enforced, **no matter the reason**, including submission issues, so be sure to submit early enough to have time to diagnose issues if problems arise.