# Homework 4

*Your name and student ID*

*Today's date*

- Due date: Tuesday, February 25 10:00pm.
- Late penalty: 50% late penalty if submitted within 24 hours of due date, no marks for assignments submitted thereafter.
- This assignment is marked out of 29. Marks are indicated for each question.
- Remember: autograder is meant as sanity check ONLY. It will not tell you if you have the correct answer. It will tell you if you are in the ball park of the answer so *CHECK YOUR WORK*
- 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 right 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. When it doesn't, 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.

**[12 points] Part 1: Simulating birth defect data and sampling from an infinitely large population**

The Center for Disease Control and Prevention (CDC) estimates that 1 in every 33 infants is born with a birth defect in the United States each year.

1) [3 points] Define a random variable for "birth defect". Write down the probability model for the random variable. Round each percentage to two decimal places (e.g., 0.43224 would be rounded to 43.22%). Is the sample space discrete or continuous?

<TODO: Replace the text with your list of proportions and your explanation.>

You might want to use the table template below to write out your probability model. If not, then delete it. *Knit now* to see how this table is rendered in your PDF.

| Tables | Are | Cool |
|---|---|---|
| yadi | type stuff | X |
| yadi | more stuff | Y |
| yada | etc | Z |

2) [2 points] Simulate data that equals 0 if there is no birth defect and equals 1 if there is a birth defect. Simulate this data for 200 births at a local hospital. Be sure to use the risk of birth defect from part a). Assign your simulated output the name `sim_01`. Print your simulated births to the screen.

Before you run your simulation, we will "set the seed". We all will set the seed to 100. This means that everyone's simulation will yield the exact same dataset.

```
set.seed(100)
# execute this line before you write your simulation code.
# only execute the set.seed() function one time.
```

```
sim_01 <- "YOUR ANSWER HERE"
sim_01
```

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

```
check_problem2()
```

```
## [1] "Checkpoint 1 Error: sim_01 should be a integer vector"
## [1] "Checkpoint 2 Error: Incorrect number of elements."
## [1] "Checkpoint 3 Error: Is seed set to 100 before running code?"
##
## Problem 2
## Checkpoints Passed: 0
## Checkpoints Errored: 3
## 0% passed
## --------
## Test: FAILED
```

Notice that `sim_01` is not a data frame, rather it is a vector of numbers. The following code stores `sim_01` as a data frame and changes its variable name. Run this code and view `sim_01` in the Viewer pane.

```
library(dplyr)
sim_01 <- as.data.frame(sim_01) # watch what happens to sim_01 in your environment
names(sim_01) # prints the variable names in the sim_01 data frame
```

```
## [1] "sim_01"
```

```
sim_01 <- sim_01 %>% rename(birth_defect = sim_01)
```

3) [2 points] Write code to determine the number of birth defects that occurred in your simulation, and the corresponding proportion with birth defects. Assign your output the name `output_01`. Print `output_01` to the screen. Hint: Use `dplyr` functions to do this.

```
output_01 <- "YOUR ANSWER HERE"
output_01
```

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

```
check_problem3()
```

```
## [1] "Checkpoint 1 Error: output_01 should be a dataframe"
## [1] "Checkpoint 2 Error: ouput_01 should have two columns."
## [1] "Checkpoint 3 Error: ouput_01 should have one row."
## [1] "Checkpoint 4 Error: Failed checkpoint for sum."
## [1] "Checkpoint 5 Error: Failed checkpoint for proportion."
##
## Problem 3
## Checkpoints Passed: 0
## Checkpoints Errored: 5
## 0% passed
## --------
## Test: FAILED
```

4) [2 marks] Re-run your simulation four more times and assign the output to a unique name each time. Print to the screen the number and proportion after each run. (Basically, "recycle" the code above four times)

```
sim_02 <- "YOUR ANSWER HERE"
output_02 <- "YOUR ANSWER HERE"
output_02
```

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

```
sim_03 <- "YOUR ANSWER HERE"
output_03 <- "YOUR ANSWER HERE"
output_03
```

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

```
sim_04 <- "YOUR ANSWER HERE"
output_04 <- "YOUR ANSWER HERE"
output_04
```

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

```
sim_05 <- "YOUR ANSWER HERE"
output_05 <- "YOUR ANSWER HERE"
output_05
```

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

```
check_problem4()
```

```
## [1] "Checkpoint 1 Error: Did not pass for output_02"
## [1] "Checkpoint 2 Error: Did not pass for output_03"
## [1] "Checkpoint 3 Error: Did not pass for output_04"
## [1] "Checkpoint 4 Error: Did not pass for output_05"
##
## Problem 4
## Checkpoints Passed: 0
## Checkpoints Errored: 4
## 0% passed
## --------
## Test: FAILED
```

5) [1 mark] Assign the vector p5 to the simulated proportions from each of your five simulation in *increasing* order.

```
p5 <- c("YOUR ANSWER HERE")
p5
```

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

```
check_problem5()
```

```
## [1] "Checkpoint 1 Error: You need to make a numeric vector"
## [1] "Checkpoint 2 Error: Need to input 5 values"
##
## Problem 5
## Checkpoints Passed: 0
## Checkpoints Errored: 2
## 0% passed
## --------
## Test: FAILED
```

6) [1 mark] Did you get close to the true value? Explain why there is variation in the proportions across the simulations.

<TODO: YOUR ANSWER HERE>

7) [1 mark] Suppose that rather than simulating 5 samples of size 200, we simulated 5 samples of size 1000. In 1-2 sentences, how would you expect the group of proportion estimates from part e) to be different? Comment both on the accuracy of these values at predicting the true value, and their variance. If you're not sure, you can re-run your simulation with a larger sample size and see how the results change to deduct the difference.

<TODO: YOUR ANSWER HERE>

**[8 points] Part 2: Probability of HIV and Hepatitis C**

Approximately 1.1 million Americans have HIV and 3.5 million Americans have Hepatitis C (HCV). The number of individuals with coinfection (e.g., both HIV and HCV) is 300,000. Among individuals with HIV, approximately 25% have Hepatitis C. The total US population was approximately 321 million at the time of these statistics.

references for these stats:

- https://www.cdc.gov/hiv/basics/statistics.html
- https://www.cdc.gov/media/releases/2016/p0504-hepc-mortality.html
- https://www.cdc.gov/hepatitis/populations/hiv.htm

8) [2 points] Calculate the probability that a randomly chosen American will have HIV. Calculate the probability that a randomly chosen American will have HCV. Convert to percentages and round to two decimal places. Save these values as the vector p2a with the proportion for HIV first then HCV. Don't inclue the % in your answer.

```
p8 <- c("YOUR ANSWER HERE")
p8
```

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

```
check_problem8()
```

```
## [1] "Checkpoint 1 Error: You need to make a numeric vector"
## [1] "Checkpoint 2 Error: Need to input 2 values"
##
## Problem 8
## Checkpoints Passed: 0
## Checkpoints Errored: 2
## 0% passed
## --------
## Test: FAILED
```

9) [2 points] Without using the number of co-infections provided in the question, calculate the probability that someone will have both HIV and HCV. Convert to a percent rounded to two decimal places and save to object p2b. Don't include the percent in your answer.

```
p9 <- "YOUR ANSWER HERE"
p9
```

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

```
check_problem9()
```

```
## [1] "Checkpoint 1 Error: Incorrect. Please enter a number"
## [1] "Checkpoint 2 Error: Did you round?"
##
## Problem 9
## Checkpoints Passed: 0
## Checkpoints Errored: 2
## 0% passed
## --------
## Test: FAILED
```

10) [2 points] Are HIV and HCV infections independent? Show work to support your answer. Uncomment your selection.

```
#p10 <- "independent"
#p10 <- "not independent"
check_problem10()
```

```
## [1] "Checkpoint 1 Error: Did you make a selection?"
##
## Problem 10
## Checkpoints Passed: 0
## Checkpoints Errored: 1
## 0% passed
## --------
## Test: FAILED
```

11) [2 points] In general, does P(A|B) equal P(B|A)? Calculate P(HIV | HCV) and report whether or not it is equal to P(HCV | HIV).

<TODO: YOUR ANSWER HERE>

**[9 points] part 3: Screening for lung cancer**

Background reading: Read pages 258-261 (and optionally 261-264) of Baldi & Moore, Edition 4. (For earlier editions, look for the section on diagnostic testing in medicine or on screening which covers sensitivity, specificity, negative predictive value, and positive predictive value).

Lung cancer is a leading cause of cancer-related deaths in the United States. Researchers examined the idea of testing all Medicare-enrolled heavy smokers for lung cancer with a computed tomography (CT) scan every year. In this population, the lifetime chance of developing lung cancer is high. In any given year, approximately 3% of heavy smokers develop lung cancer. The CT scan positively identifies lung cancer 89% of the time, and it gives a negative results for 93% of individuals who do not have lung cancer.

12) [3 points] Use probability notation to express the three proabilities cited. Make sure to define each event using a capital letter (or two). Provide the terminology for the 89% and 93% values based on your readings.

<TODO: YOUR ANSWER HERE>

13) [3 points] What percent of CT scans in this target population would be positive? Answer this question by making either a probability tree or using absolute frequencies. Show your work.

<TODO: YOUR ANSWER HERE>

<Note: If you are writing your solutions in R markdown you may want to upload an image of a hand-drawn tree diagram (this is optional). If so use the following code, or delete if not using. Be sure to remove the option "eval = F" if using this code or it won't run when you knit the file!:>

14) [1 point] We will now solve for the probability that a Medicare-enrolled heavy smoker who gets a positive scan actually has lung cancer. Write out the probability statement for this amount.

<TODO: YOUR ANSWER HERE>

15) [1 point] Calculate the probability value from question 14 based on your previous work from question 13. Store the answer as a percentage rounded to one decimal place in the object p15.

```
p15 <- "YOUR ANSWER HERE"
p15
```

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

```
check_problem15()
```

```
## [1] "Checkpoint 1 Error: Is your answer a number?"
## [1] "Checkpoint 2 Passed"
## [1] "Checkpoint 3 Error: Is your answer rounded?"
##
## Problem 15
## Checkpoints Passed: 1
## Checkpoints Errored: 2
## 33.33% passed
## --------
## Test: FAILED
```

16) [1 point] What term from your reading does this value represent? Store the answer in object p16.

```
p16 <- "YOUR ANSWER HERE"
p16
```

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

```
check_problem16
```

```
## function ()
## {
##     problem_num <- 16
##     max_scores[problem_num] <<- 1
##     num_tests <<- 1
##     problem_types[problem_num] <<- "autograded"
##     problem_names[problem_num] <<- sprintf("Problem %d", problem_num)
##     tests_failed <<- num_tests
##     checkpoint(checkpoint_number = 1, test = class(p16) == "character",
##         error_message = "Is your answer a character?")
##     if (tests_failed == 0 && problem_types[problem_num] != "free-response") {
##         scores[problem_num] <<- max_scores[problem_num]
##     }
##     else {
##         scores[problem_num] <<- 0
```

```
##     }
##     assert_that(tests_failed <= num_tests, tests_failed >= 0,
##         msg = sprintf("Did you set your num_test correctly for problem %d?",
##             problem_num))
##     return_score(problem_num, num_tests, tests_failed)
## }
## <bytecode: 0x0000000019418a90>
```

```
check_problem16()
```

```
## [1] "Checkpoint 1 Passed"
##
## Problem 16
## Checkpoints Passed: 1
## Checkpoints Errored: 0
## 100% passed
## --------
## Test: PASSED
```

### 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  NOT YET GRADED               3 free-response
## Problem 2           FAILED               2    autograded
## Problem 3           FAILED               2    autograded
## Problem 4           FAILED               2    autograded
## Problem 5           FAILED               1    autograded
## Problem 6  NOT YET GRADED               1 free-response
## Problem 7  NOT YET GRADED               1 free-response
## Problem 8           FAILED               2    autograded
## Problem 9           FAILED               2    autograded
## Problem 10          FAILED               2    autograded
## Problem 11 NOT YET GRADED               1 free-response
## Problem 12 NOT YET GRADED               3 free-response
## Problem 13 NOT YET GRADED               3 free-response
## Problem 14 NOT YET GRADED               1 free-response
## Problem 15          FAILED               1    autograded
## Problem 16          PASSED               1    autograded
```

**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-sp20/hw/hw04; 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.