Studio 7 Significance testing and the probability of hypotheses
18.05, Spring 2022

Overview of the studio

In this studio we will be frequentists. But secretly we will know the actual probability of the different hypotheses. This will illustrate how frequentist values like significance and power are NOT probabilities that a hypothesis is true. They are probabililities **assuming** a specific hypothesis is true. That is, they are likelihoods.

More specifically, we will simulate tossing a coin with unknown probability of heads and testing whether the coin is fair or not. We'll call θ the unknown probability of heads. In each problem we will have the following:

1. H_0: The coin is fair. In each problem, we will give the probability of head given H_0: `theta_H0 = 0.5`

2. H_A: The coin has has a specified probability of heads `theta_HA`. `theta_HA` will be given as an argument to the problem functions.

3. We will use a secret prior, named `secret_prior`, so we can see how various error rates change with the prior.

The take-away is that numbers like significance give you error rates conditioned on a specific hypothesis. They do not give absolute error rates. That requires a prior which is not available to the Frequentist.

R introduced in this studio

See the sample code for examples.

We will use `qbinom()` to find rejection region boundaries (critical values). Because the distribution is discrete, it is easy to make an 'off-by-one' error. The sample code shows how to take care to avoid this.

We use the `%in%` operator to check if one value is contained in a list.

We review giving the `sample` function a prescribed sampling distribution.

Download the zip file

- You should have downloaded the file `mit18_05_s22_studio7.zip` from our MITx site.
- Unzip it in your 18.05 studio folder.
- You should see the following R files
  `mit18_05_s22_studio7.r`
  `mit18_05_s22_studio7-samplecode.r`
  `mit18_05_s22_studio7-test.r`

  and the following other files

`mit18_05_s22_studio7-test-answers.html`

Prepping R Studio

- In R studio, open `mit18_05_s22_studio7-samplecode.r` and `mit18_05_s22_studio7.r`
- Using the Session menu, set the working directory to source file location. (This is a good habit to develop!)
- Answer the questions in the detailed instructions just below. Your answers should be put in `mit18_05_s22_studio7.r`
- Solution code will be posted tomorrow at 10 pm

Detailed instructions for the studio

• Go through **mit18_05_s22_studio7-samplecode.r** as a tutorial.

Right-sided rejection regions

To avoid too much fussy coding, we will always require that `theta_HA > 0.5`. This means we will run a **right-sided significance test** with significance level alpha. The code we provide will enforce this.

You should understand why that, if `theta_HA < 0.5`, we would need to do a left-sided test.

Problem 1

Problem 1. Here you will finish the code for the function

`studio7_problem_1(theta_HA, alpha, n_tosses)`

The arguments to this function are:
`theta_HA` = probability of heads for an unfair coin
`alpha` = the significance level for the test
`n_tosses` = the number of coin tosses in one trial

For this problem you are omniscient. You always know the answer. It's just the poor experimenters below you who don't know if they are making errors by rejecting or not rejecting.

There are two types of coins with 0.5 and `theta_HA` probability of heads. An experimenter has one of the coins and wants to determine which type it is.

Null hypotheses H0: the coin is fair, `theta_H0 = 0.5`.
Alternative hypotheses HA: the coin has probability `theta_HA` of heads.

Experiment: toss the coin n_tosses times

Test statistic: x = number of heads

For this problem, your code needs to determine the rejection region, actual significance for this rejection region and power* of this test.

Your code should print out these values.

*The power is defined as $P(\text{rejection} \mid H_A)$, i.e. the probability of a true positive.

Problem 2

Problem 2. Here you will finish the code for the function
`studio7_problem_2(theta_HA, alpha, n_tosses, n_trials, secret_prior)`

The arguments to this function are:

`theta_HA` = probability of heads for an unfair coin
`alpha` = the significance level for the test
`n_tosses` = the number of coin tosses in one trial
`n_trials` = the number of trials to run in the simulation
`secret_prior` = the secret prior used to pick the type of coin for each trial, i.e.
`secret_prior = c(probability_of_H0, probability_of_HA)`

As before H_0 = the coin is fair, H_A = the coin has probability theta_HA of heads.

One trial consists of:
1. Choosing a type of coin using the secret_prior.

2. Tossing it n_tosses times and counting the number of heads. The tosses should use the probability appropriate to the type of coin chosen in step 1.

3. Deciding whether or not to reject H_0.

Run n_trials trials and keep track of

the number of rejections, the number of type 1 errors, the number of type 2 errors

number of times H0 is chosen, number of times HA is chosen

(You can only track errors and the true state of nature because you are omniscient. In real life researchers can't do this.)

Use your results to estimate each of the following

$$P(\text{rejection} \mid H_0), \quad P(H_0 \mid \text{rejection}), \quad P(\text{rejection} \mid H_A), \quad P(H_A \mid \text{rejection})$$

Then print out all of the following information: (see the test answers file for what the output should look like.)

- theta_HA, alpha, n_tosses, n_trials, secret_prior
- number of rejections
- number of type 1 errors
- number of type 2 errors
- estimated $P(\text{rejection} \mid H_0)$
- estimated $P(H_0 \mid \text{rejection})$
- estimated $P(\text{rejection} \mid H_A)$
- estimated $P(H_A \mid \text{rejection})$

- estimated P(rejection)

Problem 3

In problem 3 you will use your function for problem 2 to explore the difference between P(hypothesis | rejection) and P(rejection | hypothesis).

Problem 3a. Here you will finish the code for the function
`studio7_problem_3a(theta_HA, alpha, n_tosses, n_trials)`

The arguments to this function are the same as in Problem 2, except they don't include `secret_prior`.

For this problem, the experiement is run with only fair coins, i.e.

`secret_prior = c(1.0, 0.0)` .

The code provided specifies the secret prior and calls studio7_problem2() for you.

First, call this function with various values of its arguments. Look at the results.

Your job is to modify the cat statements at the end of this function to print a short explanation of the estimated probabilities that go with the given prior.

Problem 3b. Here you will finish the code for the function
`studio7_problem_3b(theta_HA, alpha, n_tosses, n_trials)`

This problem is identical to 3a, except the prior is set to only produce unfair coins, i.e.

`secret_prior = c(0.0, 1.0)` .

Problem 3c. Use the function
`studio7_problem_3c()`
to print out a few lines explaining the difference between $P(H_0 \,|\, \text{rejection})$ and significance.

Problem 3d. Use the function
`studio7_problem_3d()`
to have R shout 5 times in all caps, "THE SIGNIFICANCE IS NOT THE PROBABILITY OF AN ERROR GIVEN REJECTION!"

Then have it more calmly state once, "It is the probability of rejection given H0. Frequentists don't compute P(Error | rejection)"

OPTIONAL Problem 4

Optional Problem 4. Here you will finish the code for the function
`studio7_problem_4(theta_HA, alpha, n_tosses, prior)`

The arguments to this function have the same meaning as in the problems above.

In this problem we will be open Bayesians and use the given prior to compute

$$\text{The true value of } P(H_0 \,|\, \text{rejection})$$
$$\text{The true value of } P(H_A \,|\, \text{rejection})$$

Testing your code

For each problem, we ran the problem function with certain parameters. You can see the function call and the output in `mit18_05_s22_studio7-test-answers.html`. If you call the same function with the same parameters, you should get the same results as in `mit18_05_s22_studio7-test-answers.html` – if there is randomness involved the answers should be close but not identical.

For your convenience, the file `mit18_05_s22_studio7-test.r` contains all the function calls used to make `mit18_05_s22_studio7-test-answers.html`.

Before uploading your code

1. Make sure all your code is in `mit18_05_s22_studio7.r`. Also make sure it is all inside the functions for the problems.
2. Clean the environment and plots window.
3. Source the file.
4. Call each of the problem functions with the same parameters as the test file `mit18_05_s22_studio7-test-answers.html`.
5. Make sure it runs without error and outputs just the answers asked for in the questions.
6. Compare the output to the answers given in `mit18_05_s22_studio7-test-answers.html`.

Upload your code

Use the upload link on our MITx site to upload your code for grading.

Leave the file name as `mit18_05_s22_studio7.r`. (The upload script will automatically add your name and a timestamp to the file.)

You can upload more than once. We will grade the last file you upload.

Due date

Due date: The goal is to upload your work by the end of class.

If you need extra time, you can upload your work any time before 10 PM ET the day after the studio.

Solutions uploaded: Solution code will be posted on MITx at 10 PM the day after the studio.

18.05 Introduction to Probability and Statistics
Spring 2022