Module 4: The Big Doc of Example Code

STA130 teaching team

Contents

This is a Big Doc of Example Code! Make sure you're working to understand the code you're running, but we hope this helps you.

Running hypothesis tests in R

All of our tests have a similar overall structure, here is some 'pseudo-code' (i.e., it won't really run) that outlines the structure you want each time.

1. State your hypotheses

You should have thought this through before you touch any code! If you want to nicely format there is R Markdown, you set up a 'math environment' with a pair of opening and closing single or double dollar signs ($). A pair of single dollar signs will put your formatted output in line with the rest of your text, while a pair of double dollar signs will centre your output and put it on a new line.

For example, if I want $H_0 : p = 0.5$, I type `$H_0: p = 0.5$`.

Or it can be on a new line and centred, if you use the double dollar signs:

$$H_0 : p = 0.5$$

$$H_0 : p \neq 0.5$$

(The above was typed as `$$H_0: p = 0.5$$ $$H_A: p \ne 0.5$$`.)

You can get the 'not equals to' sign by typing `\ne`

Make sure you also explain in words what each of your hypotheses actually mean!

Symbols review

Statistic type	Population	Sample	In R
proportion	p	$\hat{p}$	p and $\hat{p}$
mean	μ	$\bar{x}$	μ and $\bar{x}$
median	'median' (no common symbol used)	$\tilde{x}$	$median$ and $\tilde{x}$

2. Calculate your test statistic (real world)

2A. Load `tidyverse` and data

Make sure you have the `tidyverse` loaded, and whatever dataset you are using, if there is one. Sometimes you might not have data, and instead just have some information that was reported, like "in the sample, 10 of 21 flips were heads". You might also like to use some of the skills you've already learned to visualize and explore the data (not shown here).

```
library(tidyverse)
my_real_data <- read_csv("my_real_data.csv") # if there is data & it is in a CSV
```

2B. Save an object that says how many observations are in our sample data

This is our real world sample, and we'll we'll use this to guide our simulation (in some examples cases). You might not always use this directly in your simulation, but you should always know it anways.

```
n_observations <- 400
```

2C. Calculate the test statistic

This could be a proportion, a difference of means, a difference of medians, a difference of proportions, and lots more! That said, these are the main ones we'll talk about in this module

You will likely need to draw on some of the summarizing and mutating you learned in Module 3 to calculate this.

```
test_stat <- ________ #some calculation using my real world data>
test_stat
```

3. Simulate under the null hypothesis

3A. Set values for simulation

We set our seed so that we get the same random starting point each time. You won't be asked to explain this function but if you don't run this with your simulation code you might keep getting slightly different answers, which will make writing up your comments very hard. Tip: Make sure you follow any instructions given for what your personal set seed number should be.

`repetitions` is how many simulations you want R to conduct. 1000 is a good number of repetitions if you don't have a reason to pick something else

We'll use `simulated_stats` to store our simulated statistics later (we'll make 1000 simulated statistics if we've set repetitions to 1000).

```
# Suppose the last three digits of my student ID were 123 and I was asked
# to use the last three digits for my set seed
set.seed(123)
repetitions <- 1000
simulated_stats <- rep(NA, repetitions)
```

3B. Automate simulation with a `for` loop (simulation world)

This is the most important step and usually requires the most careful thought. How can you simulate data that would match your null hypothesis? Remember, in simulation world, we assume the null hypothesis is true. Once you write the code to create your new simualted data (`new_sim`), you can then calculate your simulation statistic. You'll calculate this the same way you calculated your test statistic BUT using your simulated data, not your sample data from the real world.

A `for` loop will go through and follow your instructions FOR each of the values in its set. The code `1:repetitions` will actually create a vector of all the numbers between 1 and 1000.Try `head(1:repetitions)` or `tail(1:repetitions)` to check this. So the code below says something like "for each number from 1 to 1000, create a new simulated dataset based on assuming your null hypothesis is true, then calculate the statistic you're interested in from this simulated dataset, and then store this value in the storage vector, at the location that matches the repetition we are on". This means the

simulation statistic we get from the 7th dataset we simulate in this loop will be saved in the 7th position in our vector. For the Python folks out there, R starts at 1 not at 0 with indexing.

```
for (i in 1:repetitions){
  # this is were you create one simulation
  new_sim <- _____

  # you'll basically use whatever code you used for test_stat above,
  # but on the new_sim data/vector
  sim_val <- _____

  # store this stat in its own little slot in the storage vector we made in 3C.
  simulated_stats[i] <- sim_val
}
```

3C. Turn results into a data frame so we can use ggplot for plotting

Convert your vector of simulated statistics, produced in the substep above, into a tibble for easy plotting (first line) and then plot it. If you've done things correctly, your hypothesized value from the null hypothesis, should be in the middle of your chart.

```
sim_tibble <- tibble(simulated_statistics = simulated_stats)
```

3D. Plot simulated statistics

Note: I just like the way `theme_minimal()` changes the look of the final chart. Not required. Likewise, I outlines my bars in black becuase I like the way it looks. Lots of customizability with ggplot!

```
ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 30, colour = "black") +
  labs(x="Here is my excellent x-axis label", y="Here is a y-axis label",
        title = "Oooh, what a nice graph of my simulated statistics!") +
  theme_minimal()
```

4. Evaluate the evidence against the null hypothesis

To visualize how we get our p-value, it might help to mark on your graph the cutoffs from which you'll calculate how many of the simulated statistics are at least as far away from the null hypothesized value as our test statistic is. Below uses the same code as the graph from above, but with new vertical lines added. I've chosen to make them red and blue, but it doesn't matter.

Note: I just like the way `theme_minimal()` changes the look of the final chart. Not required. Likewise, I outlines my bars in black becuase I like the way it looks. Lots of customizability with ggplot!

```
# the value from your null hypothesis
hypothesized_value <- _____

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 30, colour = "black") +
  labs(x="Here is my excellent x-axis label", y="Here is a y-axis label",
        title = "Oooh, what a nice graph of my simulated statistics!") +
```

```
  geom_vline(xintercept = hypothesized_value - abs(test_stat-hypothesized_value),
             colour = "red") +
  geom_vline(xintercept = hypothesized_value + abs(test_stat-hypothesized_value),
             colour = "blue") +
  theme_minimal()
```

How do I get the code above to make the lines? To mark the points that are at least as far from my hypothesized value as my test statistic, I need to first find that distance, which will be: `test_stat-hypothesized_value` and then taking the absolute value with the `abs()` function will just ensure this value is positive (think of it as just distance, not direction). Then I can just start at my hypothesized value and walk to the left the required distance (`hypothesized_value - abs(test_stat-hypothesized_value)`) and mark it, and then walk to the right (starting at the hypothesized value) (`hypothesized_value + abs(test_stat-hypothesized_value)`), and mark that. The number of observations that are at this cutoff or more extreme (the far left or the far right of the chart) give us the number of times, out of 1000, that we got a value like our test statistic, or more extreme, when the null hypothesis was true. We can use this in our p-value calculation, too!

Calculate your p-value by keeping only the simulated statistics that were like our test statistic or more extreme.

```
p_value <- sim_tibble %>%
  filter(simulated_statistics <= hypothesized_value - abs(test_stat-hypothesized_value) |
         simulated_statistics >= hypothesized_value + abs(test_stat-hypothesized_value)) %>%
  summarise(p_value = n() / repetitions) %>%
  as.numeric()
p_value
```

5. Make a conclusion

This part is all about putting everything together and writing up your results.

Some guidelines for how small is small? This table tells you how to comment on the **strength of evidence against** H_0.

P-value	Evidence
p-value > 0.10	no evidence against H_0
$0.05 <$ p-value < 0.10	weak evidence against H_0
$0.01 <$ p-value < 0.05	moderate evidence against H_0
$0.001 <$ p-value < 0.01	strong evidence against H_0
p-value < 0.001	very strong evidence against H_0

Sometimes, you'll see some conclusions talking about *statistical significance* (or a *statistically significance* difference)

- A significance level (α) set in advance determines the cut-off for how unusual/extreme the test statistic has to be (assuming H_0 is true) in order to reject the assumption that H_0 is true (i.e. to conclude statistical significance)
- α can be chosen to be any number, but typically $\alpha = 0.05$

It is **better** to report the p-value and comment on the *strength of evidence against* H_0 instead of only reporting whether the result is/isn't statistically significant

But, sometimes, we apply a rule like "Reject H_0 if p-value $\leq \alpha$".

Creating a simulated dataset

Simulating basic probability tools, like cards, coins and dice

You'll use an approach like this when you are doing a hypothesis test for just one group.

Example questions would be:

- Is the coin fair, i.e. is p = 0.5?
- Does the number 5 have a 1 in 6 chance of being rolled on this dice?

You will set up a vector, `x`, of the possible values and sample from it, with replacement for a size equal to your observed data. You must sample with replacement. We only need to set `prob` if we want to specify a probability of being pulled that is different to the proportion of times that option appears in `x`. This would be most common if we were simulating something like '5' or 'not 5' on a dice.

```
# Is the coin fair, i.e. is p = 0.5? This is what we'd simulate:
new_sim <- sample(x = c("Head","Tail"), size=20, replace=TRUE)

# Does the number 5 have a 1 in 6 chance of being rolled on this dice?
# This is what we'd simulate:
new_sim <- sample(x = c("5","Not 5"), size=20, replace=TRUE, prob = c(1/6, 5/6))
```

Simulating 'shuffling labels'

You'll use an approach like this when you are doing a hypothesis test for two independent groups. Two groups are independent if what happens/is observed in one group is unrelated to what happens/is observed in the other group. You'll learn more about statistical independence in second-year statistics courses.

What we are trying to do with this is simulate the hypothesis that the group/treatment has nothing to do with the other variable we've observed, that each person/unit would have had the same result, even if they had been assigned to a different group. So, what we do, is simulate just randomly re-assigning everyone to a group. This aims to mimic the way the initial experiment was run, BUT is in simulation world. We are not running a new experiment when we do this.

```
# Start with your real data, and then randomly reassign the group labels
# This overwrites the original group labels
new_sim <- my_real_data %>% mutate(label = sample(label))
```

Example 1: one group, proportion

1. State hypotheses

In the Wheel of Destiny problem, we started with a question: "Is the new Wheel of Destiny fair?"

Two hypotheses:

Null hypothesis (H_0): The Wheel of Destiny spinner is fair

$$H_0 : p_{red} = 0.5$$

Alternative hypothesis (H_A or H_a or H_1): The Wheel of Destiny spinner is not fair

$$H_1 : p_{red} \neq 0.5$$

where p_{red} is the proportion of spins of the new Wheel of Destiny that land on red (if we spun it infinitely many times).

2. Calculate the test statistic from observed data

```
test_stat <- 32/50
test_stat
```

```
## [1] 0.64
```

3. Simulate samples assuming H_0 is true and calculate the statistic for each sample

Goal: Explore the distribution of values of the statistic (in this case, $\hat{p}_{red}$) we would observe if H_0 was true. What kind of results are common under the null? Which are unusual?

Simulation is a way to explore random events.

Previously, we used people-power to simulate values of $\hat{p}_{heads}$ (or equivalently $\hat{p}_{red}$) under the assumption of a 50/50 process (H_0)

We can use R to simulate values more quickly!

Simulating coin flips with R to get a sampling distribution

```
coin <- c("heads", "tails")
flips <- sample(coin,  # vector with all possible outcomes
       size = 50,            # number of values to sample
       prob = c(0.5, 0.5),   # probability of each outcome (default is equal probs)
       replace = TRUE)       # can outcomes be repeated? (default=FALSE)
```

What will the following lines of R code do?

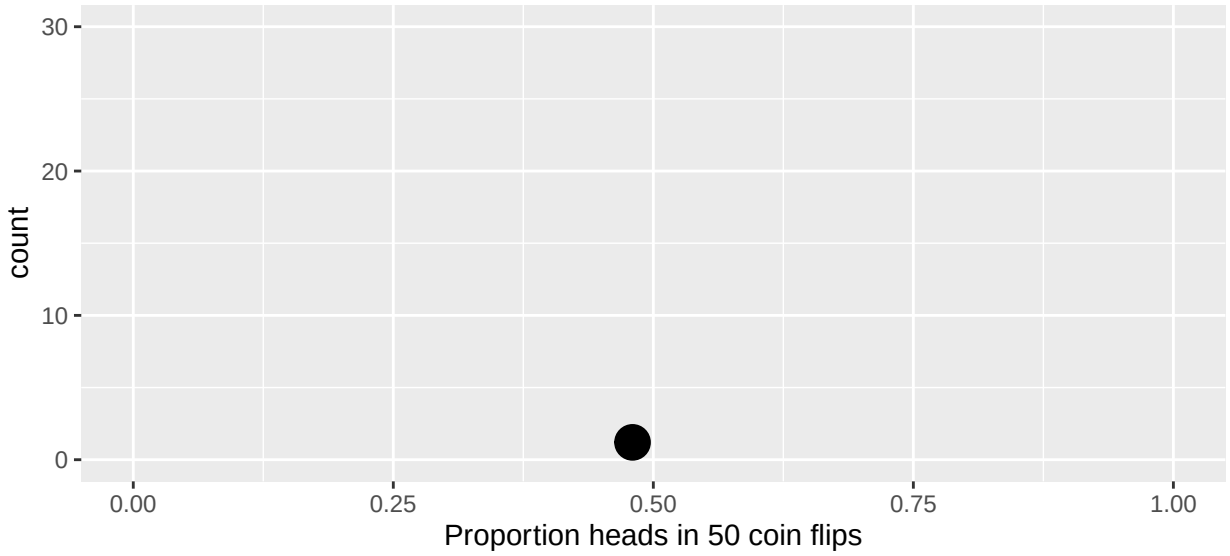
```
sum(flips == "heads")
sum(flips == "heads") / 50
mean(flips == "heads")
```

```
sim <- tibble(p_heads = mean(flips=="heads"))
print(as.numeric(sim))
```

```
## [1] 0.48
```

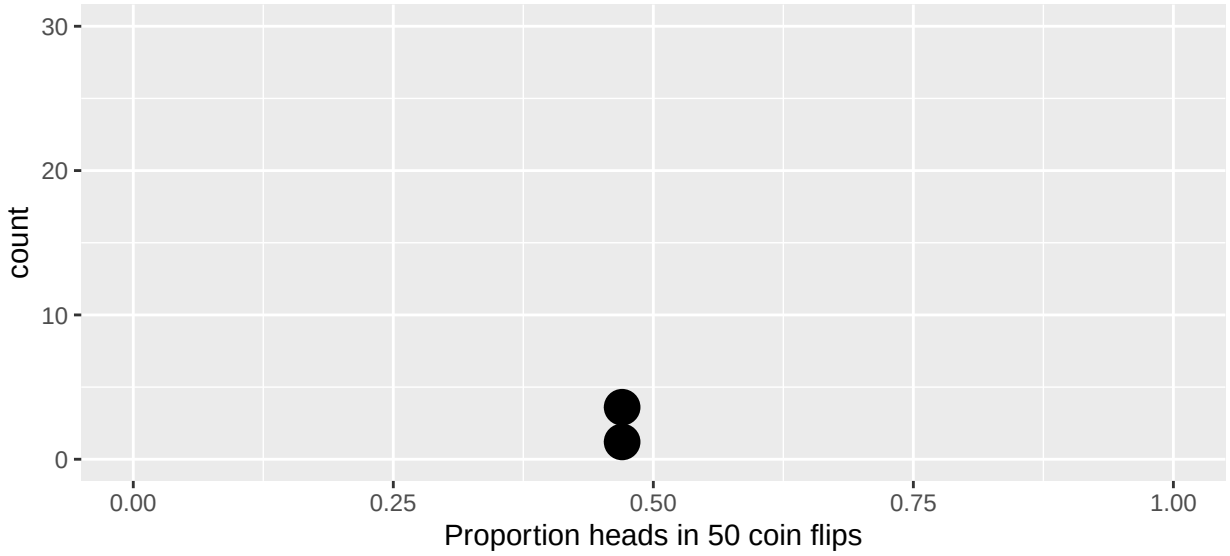
```
sim %>% ggplot(aes(x=p_heads)) +
  geom_dotplot() + xlim(0, 1) + ylim(0,30) +
  labs(x="Proportion heads in 50 coin flips")
```

```
## `stat_bindot()` using `bins = 30`. Pick better value with `binwidth`.
```



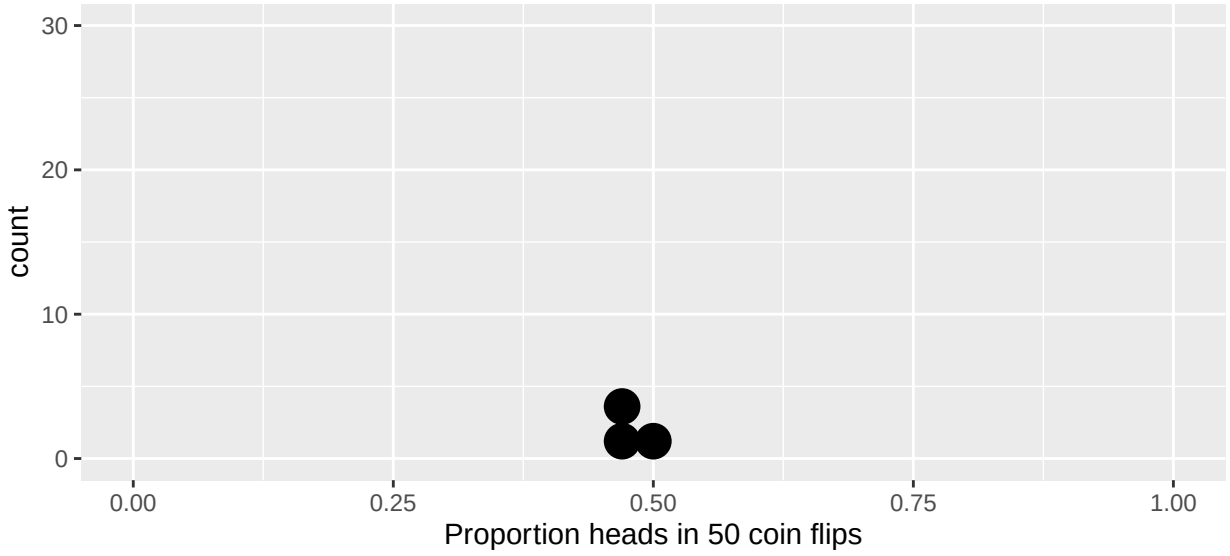
Add another simulation

```
## [1] 0.46
```



And another simulation

```
## [1] 0.5
```



We could keep going to build the sampling distribution (from simulation)... But we don't want to have to keep doing each simulation by hand!

Use `for` loops to generate many simulations

- Automate the process of generating many simulations
- Evaluate a block of code for each value of a sequence (for example, 1, 2, 3, ... 1000)
- The following `for` loop will evaluate SOME CODE 1000 times, for `i=1` and `i=2` and ... and `i=1000`
  - Note that code is within *curly* brackets

```
for (i in 1:1000){
  SOME CODE
}
```

```
n_observations <- 50 # number of observations (e.g. coin flips or spins)
repetitions <- 1000 # 1000 simulations
simulated_stats <- rep(NA, repetitions) # 1000 missing values to start
```

3A. Set values for simulation

```
for (i in 1:repetitions){
  new_sim <- sample(c("red", "black"),
                    size = n_observations,
                    prob = c(0.5, 0.5),
                    replace = TRUE)
  sim_p <- sum(new_sim == "red") / n_observations

  simulated_stats[i] <- sim_p # add new value to vector of results
}
```

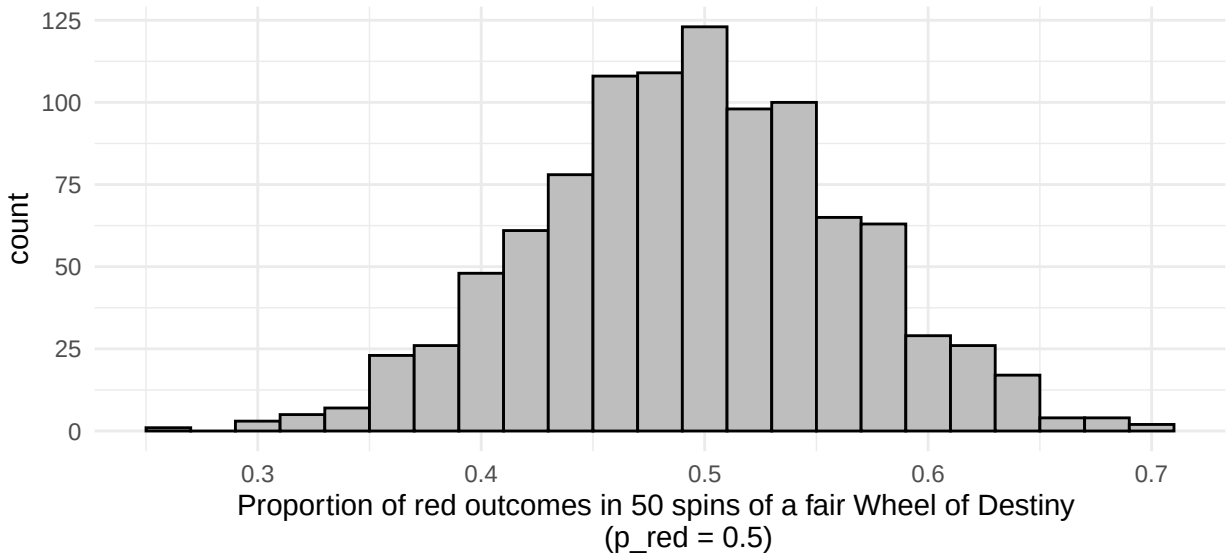
3B. Automate simulation with a `for` loop (simulation world)

```
sim_tibble <- tibble(p_red = simulated_stats)
```

3C Turn results into a data frame so we can use ggplot for plotting

3D Plot Plot simulated statistics Although samples are selected at random (so the value of the statistic is different for each sample) the distribution of all possible values of the statistic (i.e. the sampling distribution) has specific features.

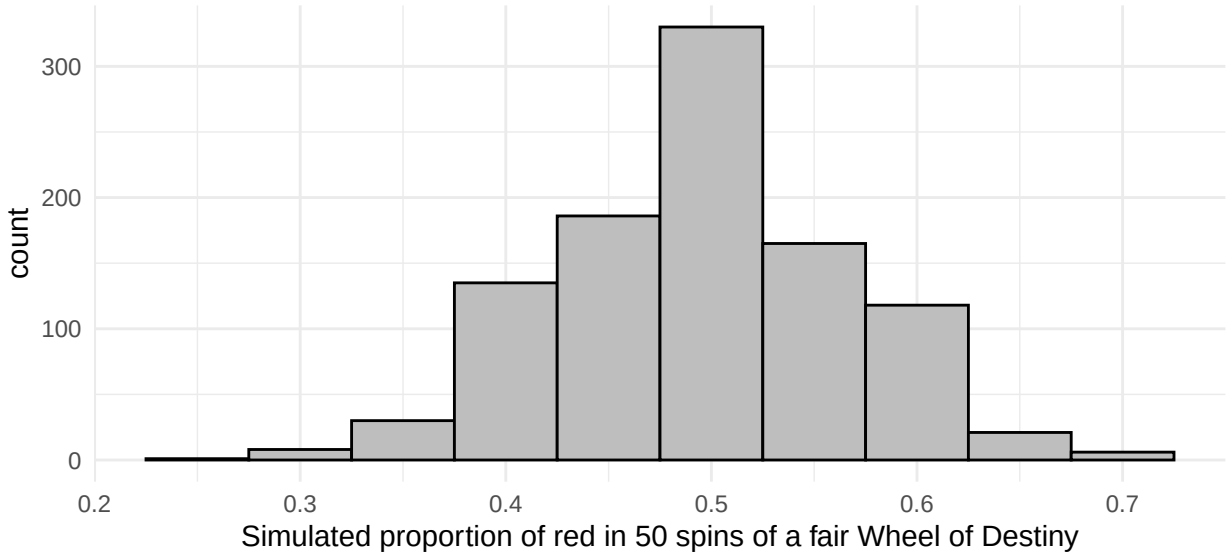
```
sim_tibble %>% ggplot(aes(x = p_red)) +
  geom_histogram(binwidth = 0.02, colour = "black", fill = "grey") +
  xlab("Proportion of red outcomes in 50 spins of a fair Wheel of Destiny
       (p_red = 0.5)") +
  theme_minimal()
```



4. Evaluate the evidence against H_0

I spun the Wheel of Destiny 50 times. In my sample (observed, not simulated!) I saw: $\hat{p}_{red} = 32/50 = 0.64$ (the proportion of our 50 spins which landed on red)

If the spinner really is fair, is our observed value of $\hat{p}_{red} = 0.64$ unusual? How unusual?



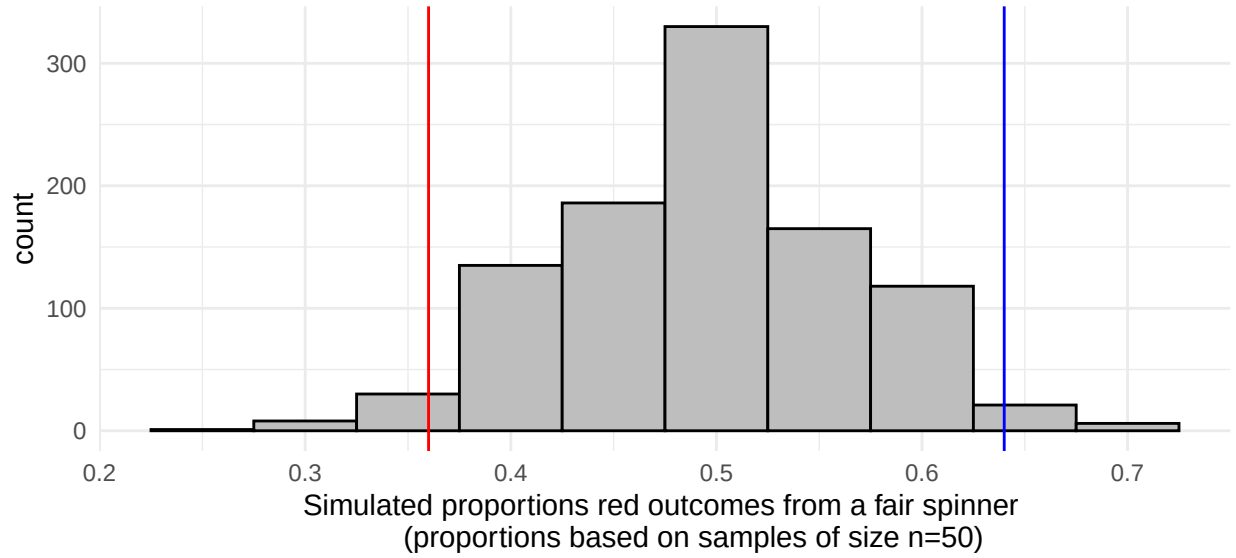
- The **p-value** is the probability of observing data that are **at least as unusual** (or **at least as extreme**) as the sample data, *under the assumption that* H_0 is true.
- We estimate the p-value as the proportion of values in the estimated sampling distribution that are as extreme or more extreme than the test statistic calculated from our observed sample data.
- For the *Wheel of Destiny* example:
  - Null hypothesis value: $p_{red} = 0.5$

  - Observed estimtate from the sample: $\hat{p}_{red} = \frac{32}{50}$
  - Values at least as extreme/unusual as the sample statistic: all values **greater or equal to** $\hat{p}_{red}$ and all values **less than or equal to** $0.5 - |\hat{p}_{red} - 0.5|$
    * i.e. values further away from the null value ($p_{red} = 0.5$) than the test statistic is (i.e. further than $|\hat{p}_{red} - 0.5|$ away from $p_{red} = 0.5$)

- This is a **two-sided test** because it considers differences from the null hypothesis that are both larger and smaller than what you observed.
  (It is also possible to carry out one-sided tests. They are useful in some specific applications.)

What proportion of simulated values are at least as far from the null value as $\hat{p}_{red}$?

```
test_stat <- 32/50

sim_tibble %>% ggplot(aes(x=p_red)) +
  geom_histogram(binwidth = 0.05, colour = "black", fill = "grey") +
  geom_vline(xintercept = 0.5  - abs(0.5 - test_stat),
             color = "red") +
  geom_vline(xintercept = 0.5 + abs(0.5 - test_stat),
             color = "blue") +
  labs(x = "Simulated proportions red outcomes from a fair spinner
       (proportions based on samples of size n=50)") +
  theme_minimal()
```




```
pvalue <- sim_tibble %>%
  filter(p_red >= 0.64 | p_red <= 0.36) %>%
  summarise(p_value = n() / repetitions)

as.numeric(pvalue)
```

```
## [1] 0.066
```

Another way to calculate the p-value is

```
p_value <- sim_tibble %>%
  filter(abs(p_red - 0.5) >= abs(test_stat - 0.5)) %>%
  summarise(p_value = n() / repetitions) %>%
  as.numeric()
p_value
```

```
## [1] 0.066
```

A small p-value tells us that there is only a small chance that we would observe a test statistic as far away from the null value of the parameter if H_0 were really true

Two reasons that can lead to a small p-value:

1. H_0 is actually true and we just observed an *unlikely extreme value* of the statistic
2. H_0 is not true

The smaller the p-value, the more we lean towards (2) - in other words, the smaller the p-value, the more "evidence" we have against H_0

5. Make a conclusion

Some guidelines for how small is small? This table tells you how to comment on the **strength of evidence against H_0**.

P-value	Evidence
p-value > 0.10	no evidence against H_0
$0.05 <$ p-value < 0.10	weak evidence against H_0
$0.01 <$ p-value < 0.05	moderate evidence against H_0
$0.001 <$ p-value < 0.01	strong evidence against H_0
p-value < 0.001	very strong evidence against H_0

Note that these are guidelines instead a hard and fast rules. (Why there aren't greater than or equal to signs.)

Conclusion (based on $\hat{p}_{red} = \frac{32}{50} = 0.64$): Since the p-value is 0.066, we conclude that we have ***weak evidence against the null hypothesis*** that the Wheel of Destiny spinner is fair.

Sometimes, you'll see some conclusions talking about *statistical significance* (or a *statistically significance* difference)

- A significance level (α) set in advance determines the cut-off for how unusual/extreme te test statistic has to be (assuming H_0 is true) in order to reject the assumption that H_0 is true (i.e. to conclude statistical significance)
- α can be chosen to be any number, but typically $\alpha = 0.05$

It is **better** to report the p-value and comment on the *strength of evidence against* H_0 instead of only reporting whether the result is/isn't statistically significant

But somtimes we apply a rule like "Reject H_0 if p-value $\leq \alpha$"

If we had picked a significance level of

$$\alpha = 0.05$$

before starting our hypothesis test, what would we conclude?

We would ***fail to reject the null hypothesis*** that the new Wheel of Destiny is fair at the $\alpha = 0.05$ significance level. In other words, we cannot reject H_0 at the $\alpha = 0.05$ significance level.

So, what advice would we give Stella? Well

Example 2: one group, proportion

At U of T, in a 2020 report, it was stated that 26.8% of our undergraduates are International students. Assume that the students in this dataset are representative of first years at their university. Is it believable that 26.8% of these students are international students?

1. State hypotheses

My null hypothesis is that the true proportion of first-year students at this university that are international students is 26.8%. My alternative is that it is not this 26.8%.

$$H_0 : p_{\text{international}} = 0.268$$
$$H_A : p_{\text{international}} \neq 0.268$$

2. Calculate the test statistic from observed data

```
library(tidyverse)
gratitude <- read_csv("gratitude.csv")
```

```
## Rows: 61 Columns: 5

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): status, residence, treatment
## dbl (2): life_satisfaction, adjustment

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Calculation version 1
test_stat <- gratitude %>%
  mutate(is_int = case_when(status == "International" ~ 1,
                            TRUE ~ 0)) %>%
  summarise(test_stat = mean(is_int)) %>%
  as.numeric()
test_stat
```

```
## [1] 0.1639344
```

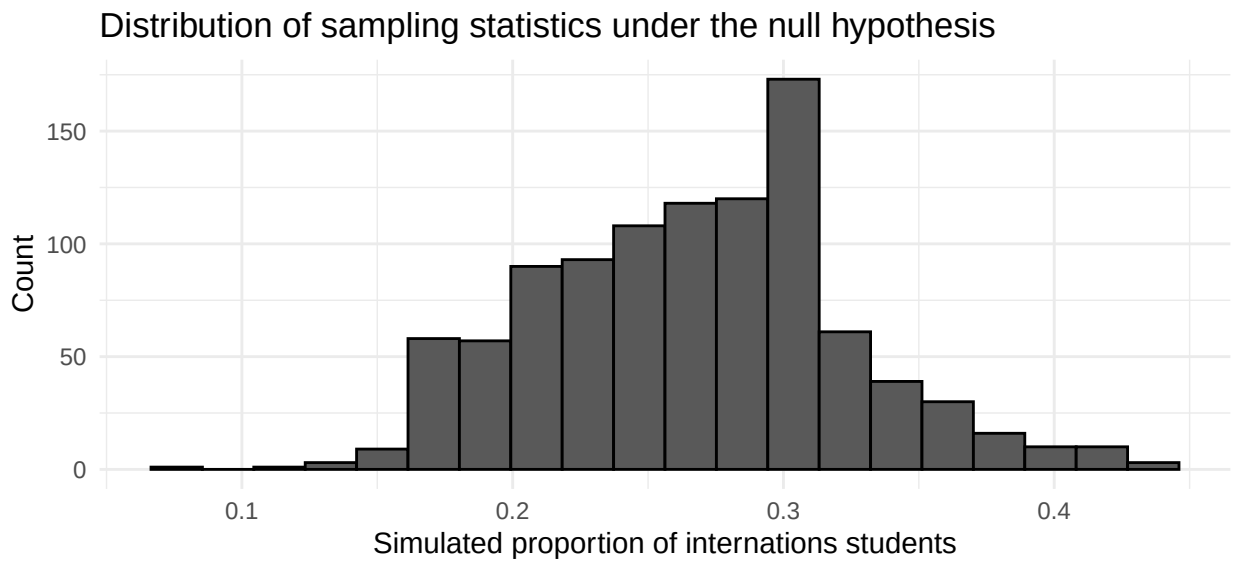
```
# Calculation version 2 (uses n_observations object)
n_observations <- nrow(gratitude)

test_stat <- gratitude %>%
  filter(status == "International") %>%
  count() %>%  # this row could also summarize(n = n())
  mutate(n = n/n_observations) %>%
  as.numeric()
test_stat
```

```
## [1] 0.1639344
```

$\hat{p} = 0.16$

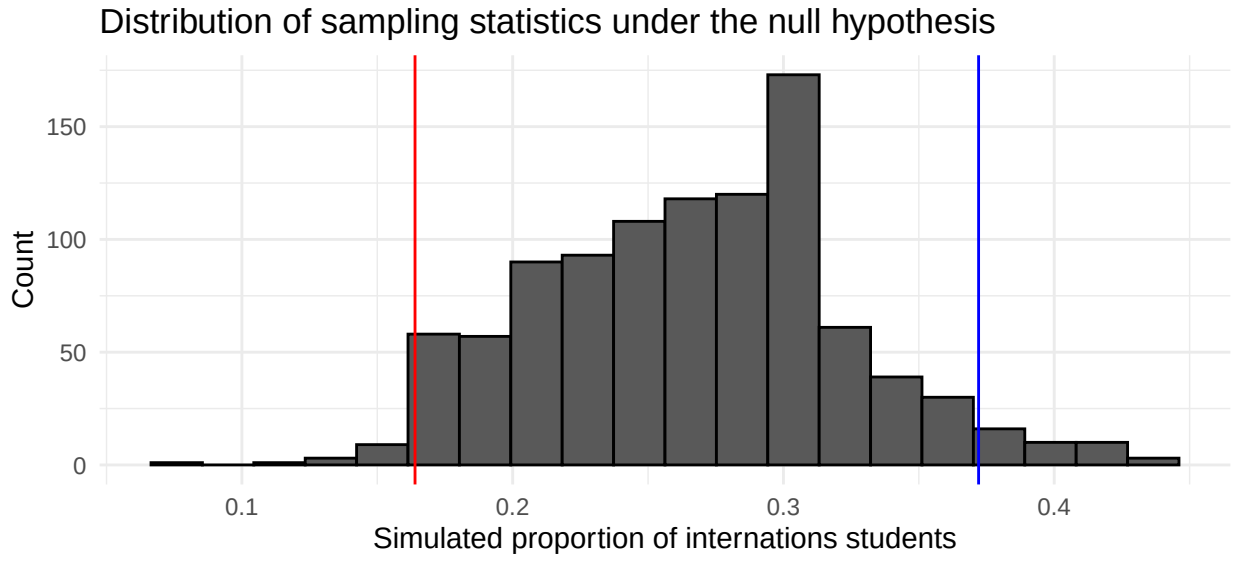
3. Simulate samples assuming H_0 is true and calculate the statistic for each sample

```
set.seed(123)

repetitions <- 1000

simulated_stats <- rep(NA, repetitions)

for (i in 1:repetitions){
  # this is were you create one simulation
  new_sim <- tibble(status = sample(c("International", "Domestic"), size = 61,
                                    replace = TRUE, prob = c(0.268, 1-0.268)))

  # you'll basically use whatever code you used for test_stat above,
  # but on the new_sim data/vector
  sim_val <- new_sim %>%
    mutate(is_int = case_when(status == "International" ~ 1,
                              TRUE ~ 0)) %>%
    summarise(test_stat = mean(is_int)) %>%
    as.numeric()

  simulated_stats[i] <- sim_val
}

sim_tibble <- tibble(simulated_statistics = simulated_stats)

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 20, color = "black") +
  labs(x="Simulated proportion of internations students", y="Count",
       title = "Distribution of sampling statistics under the null hypothesis") +
  theme_minimal()
```



4. Evaluate the evidence against H_0

```
hypothesized_value <- 0.268

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 20, color = "black") +
  labs(x="Simulated proportion of internations students", y="Count",
       title = "Distribution of sampling statistics under the null hypothesis") +
  geom_vline(xintercept = hypothesized_value - abs(test_stat-hypothesized_value),
             colour = "red") +
  geom_vline(xintercept = hypothesized_value + abs(test_stat-hypothesized_value),
             colour = "blue") +
  theme_minimal()
```



```
p_value <- sim_tibble %>%
  filter(simulated_statistics <= hypothesized_value - abs(test_stat-hypothesized_value) |
         simulated_statistics >= hypothesized_value + abs(test_stat-hypothesized_value)) %>%
  summarise(p_value = n() / repetitions) %>%
  as.numeric()
p_value
```

```
## [1] 0.073
```

5. Make a conclusion

Our p-value is 0.073. This means we have weak evidence against the hypothesis that the proportion of first-year students who are international students at this Turkish university is the same as the undergraduate proportion for U of T.

We would ***fail to reject the null hypothesis*** the proportion of first-year students who are international students at this Turkish university is the same as the undergraduate proportion for U of T at the $\alpha = 0.05$ significance level.

Note: If we had instead set out $\alpha = 0.10$ BEFORE we did the test, we WOULD reject the null hypothesis.

Example 3: two groups, difference of means

Does gratitude journaling make a difference to life satisfaction for students?

1. State hypotheses

My null hypothesis is that there is no difference in average life satisfaction scores between the treatment and control groups.

$$H_0 : \mu_{\text{treatment}} - \mu_{\text{control}} = 0$$
$$H_A : \mu_{\text{treatment}} - \mu_{\text{control}} \neq 0$$

2. Calculate the test statistic from observed data

```
# I already ran these in Example 2
# library(tidyverse)
# gratitude <- read_csv("gratitude.csv")

# don't really need to calculate the number of obs in each group this time
gratitude %>%
  group_by(treatment) %>%
  count() # count is the same as summarise(n = n())
```

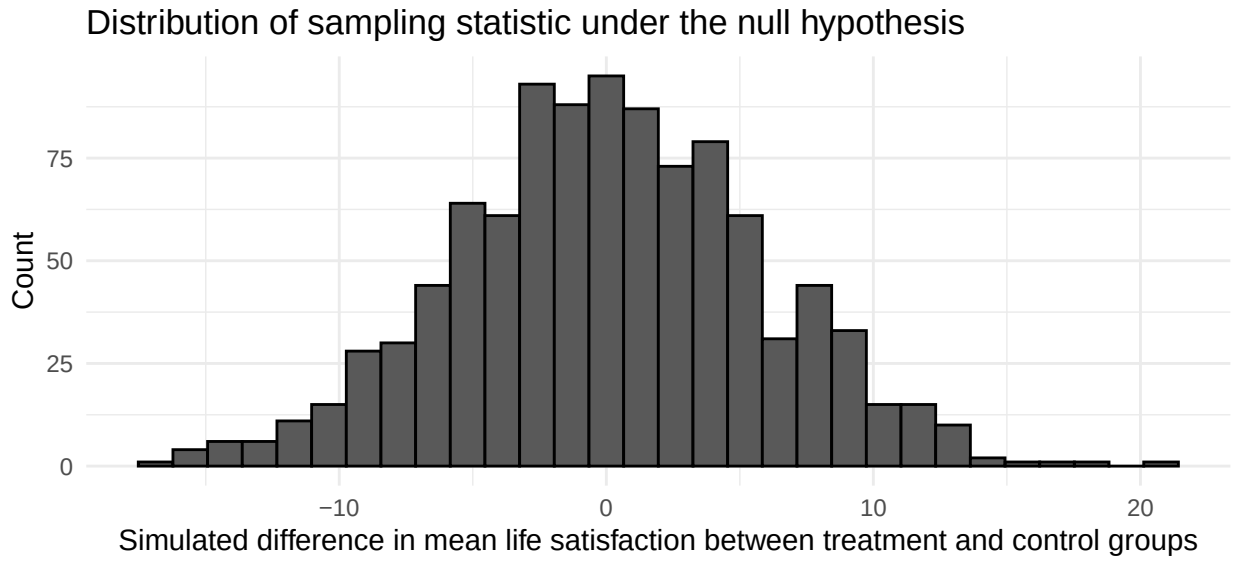
```
## # A tibble: 2 x 2
## # Groups:   treatment [2]
##   treatment     n
##   <chr>     <int>
## 1 Control      31
## 2 Treatment    30
```

```
test_stat <- gratitude %>%
  group_by(treatment) %>%
  summarise(mean = mean(life_satisfaction)) %>%
  ungroup() %>%
  summarise(test_stat = diff(mean)) %>%
  as.numeric()

test_stat
```

```
## [1] 12.53763
```

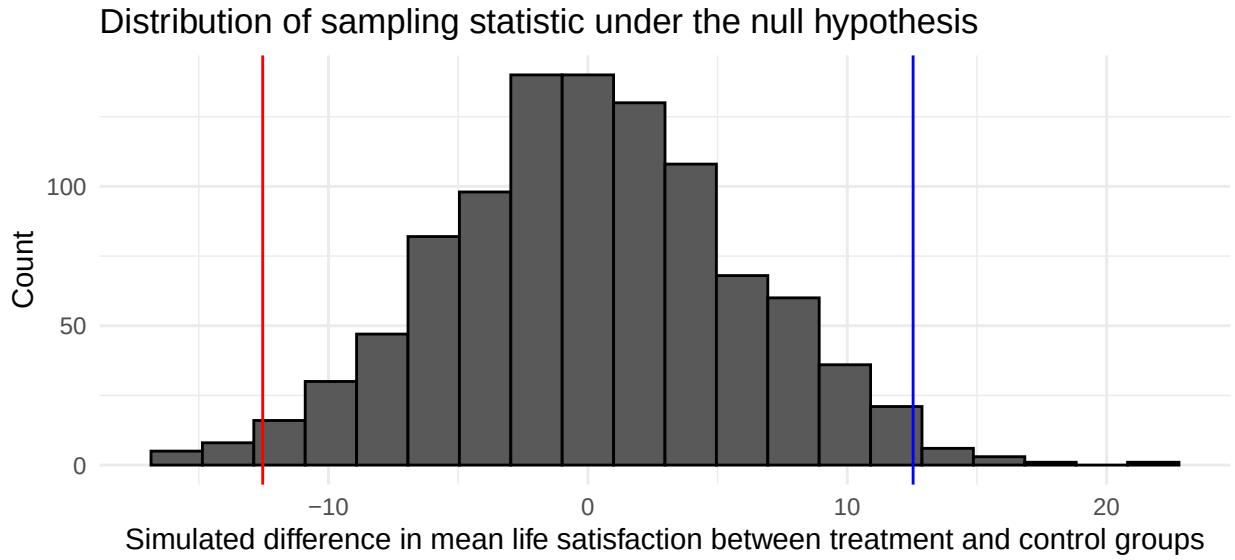
3. Simulate samples assuming H_0 is true and calculate the statistic for each sample

```r
set.seed(123)

repetitions <- 1000

simulated_stats <- rep(NA, repetitions)

for (i in 1:repetitions){
  # this is were you create one simulation
  new_sim <- gratitude %>%
    mutate(treatment = sample(treatment))

  # you'll basically use whatever code you used for test_stat above,
  # but on the new_sim data/vector
  sim_val <- new_sim %>%
   group_by(treatment) %>%
  summarise(mean = mean(life_satisfaction)) %>%
  ungroup() %>%
  summarise(test_stat = diff(mean)) %>%
  as.numeric()

  simulated_stats[i] <- sim_val
}

sim_tibble <- tibble(simulated_statistics = simulated_stats)

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 30, color = "black") +
  labs(x="Simulated difference in mean life satisfaction between treatment and control groups",
       y="Count",
       title = "Distribution of sampling statistic under the null hypothesis") +
  theme_minimal()
```



4. Evaluate the evidence against H_0

```
hypothesized_value <- 0

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 20, color = "black") +
  labs(x="Simulated difference in mean life satisfaction between treatment and control groups",
       y="Count",
       title = "Distribution of sampling statistic under the null hypothesis") +
  geom_vline(xintercept = hypothesized_value - abs(test_stat-hypothesized_value),
             colour = "red") +
  geom_vline(xintercept = hypothesized_value + abs(test_stat-hypothesized_value),
             colour = "blue") +
  theme_minimal()
```




```
p_value <- sim_tibble %>%
  filter(simulated_statistics <= hypothesized_value - abs(test_stat-hypothesized_value) |
         simulated_statistics >= hypothesized_value + abs(test_stat-hypothesized_value)) %>%
  summarise(p_value = n() / repetitions) %>%
  as.numeric()
p_value
```

```
## [1] 0.028
```

Make a conclusion

Our p-value is 0.028. This means we have moderate evidence against the hypothesis that the first-year students who gratitude journaled had the same average life satisfaction as those who didn't (the control).

We would ***reject the null hypothesis*** that the first-year students who gratitude journaled had the same average life satisfaction as those who didn't (the control) at the $\alpha = 0.05$ significance level.

Example 4: two group, diff of proportions

Is the proportion of students in university residences the same between international and domestic students?

1. State hypotheses

My null hypothesis is that the true proportion of first-year students at this university that are international students is 26.8%. My alternative is that it is not this 26.8%.

$$H_0 : p_{\text{in residence internationa}} = p_{\text{in residence domestic}}$$

$$H_A : p_{\text{in residence international}} \neq p_{\text{in residence domestic}}$$

This is equivalent to:

$$H_0 : p_{\text{in residence international}} - p_{\text{in residence domestic}} = 0$$

$$H_A : p_{\text{in residence international}} - p_{\text{in residence domestic}} \neq 0$$

Our null hypothesis is that, of first-year students at this Turkish university, the same percentage of international and domestic students stay in residence.

2. Calculate the test statistic from observed data

```
# I already ran these in Example 2
# library(tidyverse)
# gratitude <- read_csv("gratitude.csv")

# Calculation version 1
p_res_int <- gratitude %>%
  filter(status=="International") %>%
  mutate(is_int = case_when(residence == "In University residence" ~ 1,
                            TRUE ~ 0)) %>%
    summarise(test_stat = mean(is_int)) %>%
    as.numeric()

p_res_dom <- gratitude %>%
  filter(status=="Domestic") %>%
  mutate(is_int = case_when(residence == "In University residence" ~ 1,
                            TRUE ~ 0)) %>%
    summarise(test_stat = mean(is_int)) %>%
    as.numeric()


test_stat <- p_res_int - p_res_dom
test_stat
```

```
## [1] -0.1235294
```

```r
# Calculation version 2 (uses n_observations object)
n_int <- gratitude %>%
  filter(status == "International") %>%
  count() %>%
  as.numeric()

n_dom <- gratitude %>% filter(status == "Domestic") %>%  count() %>% as.numeric()

n_int_res <- gratitude %>%
  filter(status == "International" & residence == "In University residence") %>%
  count() %>%
  as.numeric()

n_dom_res <- gratitude %>%
  filter(status == "Domestic" & residence == "In University residence") %>%
  count() %>%
  as.numeric()

test_stat <- n_int_res/n_int - n_dom_res/n_dom
test_stat
```

```
## [1] -0.1235294
```

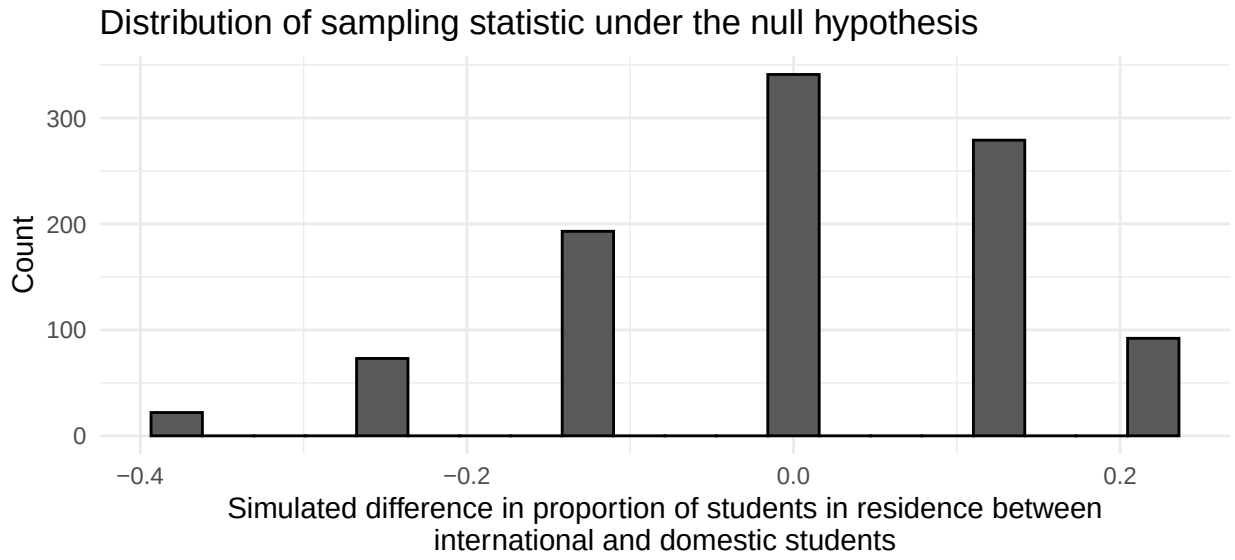
```r
# Calculation version 3 (all in one pipe)
test_stat <- gratitude %>%
  group_by(status) %>%
  mutate(total = n()) %>%
  group_by(status, residence, total) %>%
  count() %>%
  filter(residence == "In University residence") %>%
  ungroup() %>%
  summarise(test_stat = diff(n/total)) %>%
  as.numeric()
test_stat
```

```
## [1] -0.1235294
```

3. Simulate samples assuming H_0 is true and calculate the statistic for each sample

```r
set.seed(123)

repetitions <- 1000

simulated_stats <- rep(NA, repetitions)

for (i in 1:repetitions){
  # this is were you create one simulation,
  new_sim <- gratitude %>%
    mutate(status = sample(status))
```

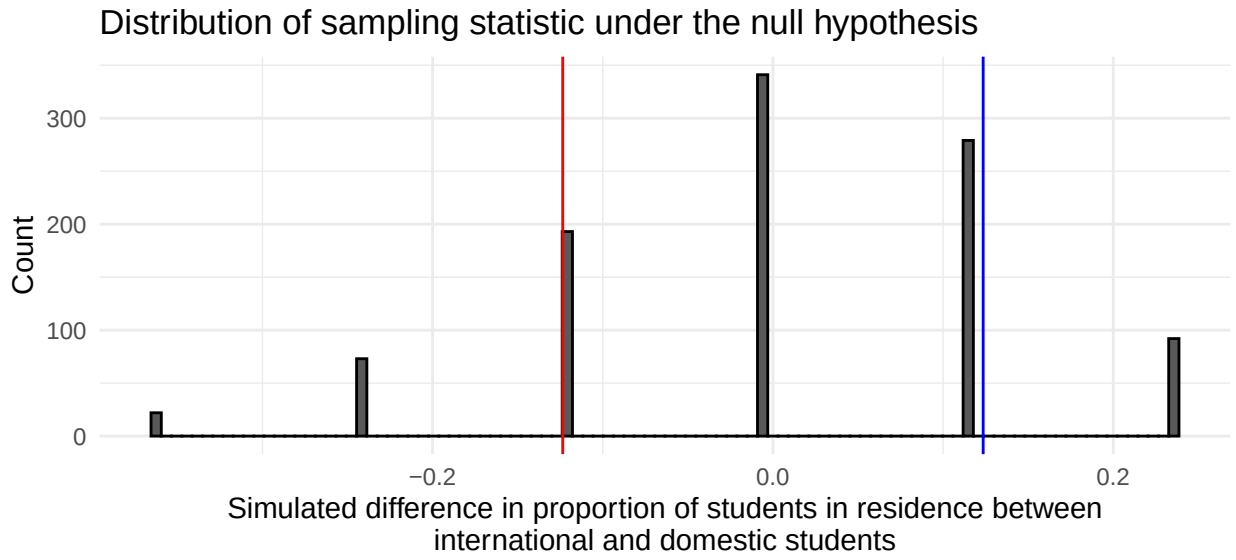
```
  # you'll basically use whatever code you used for test_stat above,
  # but on the new_sim data/vector
  sim_val <- new_sim %>%
    group_by(status) %>%
    mutate(total = n()) %>%
    group_by(status, residence, total) %>%
    count() %>%
    filter(residence == "In University residence") %>%
    ungroup() %>%
    summarise(test_stat = diff(n/total)) %>%
    as.numeric()

  simulated_stats[i] <- sim_val
}

sim_tibble <- tibble(simulated_statistics = simulated_stats)

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 20, color = "black") +
  labs(x="Simulated difference in proportion of students in residence between\ninternational and domesti
       title = "Distribution of sampling statistic under the null hypothesis") +
  theme_minimal()
```



4. Evaluate the evidence against H_0

```
hypothesized_value <- 0

ggplot(sim_tibble, aes(x = simulated_statistics)) +
  geom_histogram(bins = 100, color = "black") +
  labs(x="Simulated difference in proportion of students in residence between\ninternational and domesti
       title = "Distribution of sampling statistic under the null hypothesis") +
  geom_vline(xintercept = hypothesized_value - abs(test_stat-hypothesized_value),
```

```
            colour = "red") +
  geom_vline(xintercept = hypothesized_value + abs(test_stat-hypothesized_value),
             colour = "blue") +
  theme_minimal()
```




```
p_value <- sim_tibble %>%
  filter(simulated_statistics <= hypothesized_value - abs(test_stat-hypothesized_value) |
         simulated_statistics >= hypothesized_value + abs(test_stat-hypothesized_value)) %>%
  summarise(p_value = n() / repetitions) %>%
  as.numeric()
p_value
```

```
## [1] 0.38
```

5. Make a conclusion

Our p-value is 0.659 This means we have no evidence against the hypothesis that the proportion of first-year students who are in residence is no different between international and domestic students at this Turkish university.

We would ***fail reject the null hypothesis*** at the $\alpha = 0.05$ significance level.

Example 5: two groups, diff of means

This is the code shown in the videos for the two group tests. More commentary there. Note a slight change in the variable name: `adjust_college` to `adjustment`.

2. Calculate the test statistic

```r
gratitude %>%
  group_by(treatment) %>%
  summarise(means = mean(adjustment))
```

```
## # A tibble: 2 x 2
##   treatment means
##   <chr>     <dbl>
## 1 Control    199.
## 2 Treatment  222.
```

```r
group_means <- gratitude %>%
  group_by(treatment) %>%
  summarise(means = mean(adjustment))

# The diff function calculates the difference between
# values in a vector
diff_means <- group_means %>%
  summarise(test_stat = diff(means))

diff_means
```

```
## # A tibble: 1 x 1
##   test_stat
##       <dbl>
## 1      23.0
```

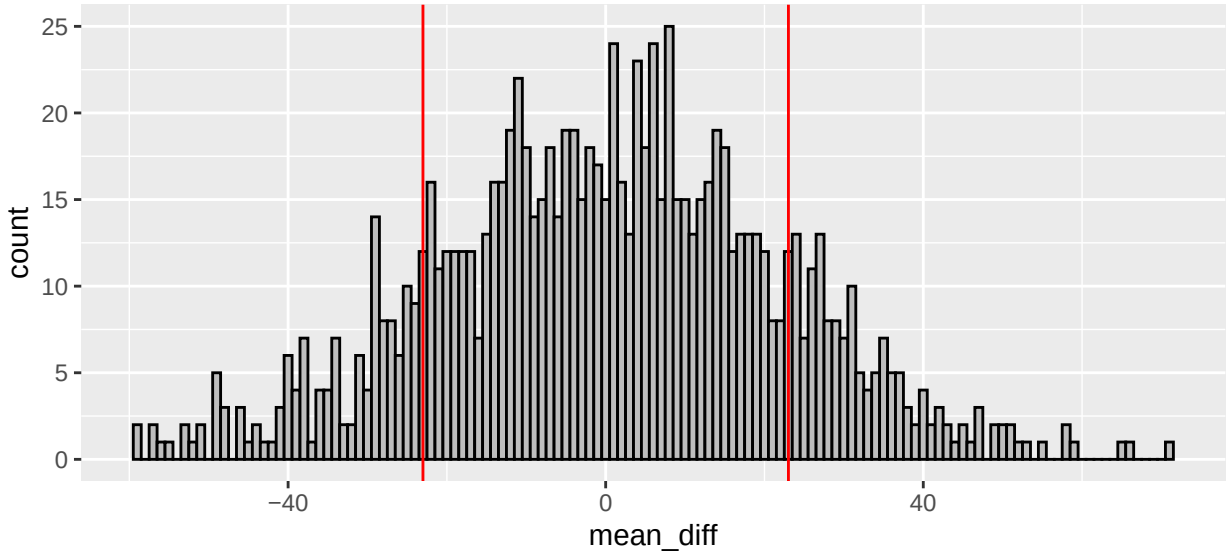
```r
as.numeric(diff_means)
```

```
## [1] 23.01505
```

3. Simulate under the H_0

```r
# Set things up so we can simulate
set.seed(523)
repetitions <- 1000
simulated_values <- rep(NA, repetitions)

# Run our simulations
for(i in 1:repetitions){
  # Here we are shuffling the labels (i.e. randomly reassigning each student
  # to a treatment group and assuming they keep the same score)
  simdata <- gratitude %>%
    mutate(treatment = sample(treatment))

  # Simulation statistic calculation,
  # exactly like our original test stat calc
  sim_value <- simdata %>%
    group_by(treatment) %>%
```

```
    summarise(means = mean(adjustment)) %>%
    summarise(value = diff(means))

  # Store your simulated statistics
  simulated_values[i] <- as.numeric(sim_value)
}

# Make into a tibble so it is easy to work with
sim <- tibble(mean_diff = simulated_values)

# Plot our estimated sampling distribution
sim %>% ggplot(aes(x=mean_diff)) +
  geom_histogram(binwidth=1, color="black", fill="gray") +
  geom_vline(xintercept = abs(diff_means$test_stat), colour = "red") +
  geom_vline(xintercept = -abs(diff_means$test_stat), colour = "red")
```



4. Evaluate evidence against the H_0

```
# Calculate p-value
num_more_extreme <- sim %>%
  filter(abs(mean_diff) >= abs(diff_means$test_stat)) %>%
  summarise(n())

p_value <- as.numeric(num_more_extreme / repetitions)
p_value
```

```
## [1] 0.292
```