# DSC 152:
# Applied Statistical Data Analysis and Inference

Lecture #1
Introduction and Background

Tuesday, March 31
Spring Quarter 2026
Peter Chi

# Who am I?

**To summarize:**

- My training and experience is heavily on the statistical side of data science
- I have many years of teaching experience at the college level, specifically statistics courses. But,
  - Prior to UCSB (in 2024), my class sizes were always much smaller (30ish max, and often single digits)
  - This specific class, DSC 152, is brand new, both to UCSD and to me!

P.S. Call me Peter!

# Who are the rest of the course staff?

# Who are you?

With any internet-capable device (including laptop, phone, tablet), please go to:

**Poll Everywhere**

https://pollev.com/chi

Hit "Skip for now" if it asks you to register for credit (I will not be tracking responses for credit).

However, please do enter your name as your screen name if you are comfortable doing so (I will be the only one who will see it).

# What is this course about?

At this point of your academic careers, you have taken:

## DSC 10: Principles of Data Science

- introductory hypothesis testing
- permutation testing
- bootstrapping

## DSC 20: Programming and Data Structures

- lots of coding, but no stats here

## DSC 30: Data Structures

- lots of coding, but no stats here

## DSC 40A: Theoretical Foundations of Data Science

- regression!

# What is this course about?

At this point of your academic careers, you have taken:

**DSC 80: Practice and Application of Data Science**

- messy/missing data
- more hypothesis testing
  - again with permutation testing and bootstrap (like in DSC 10)

**SE 125 or ECE 109 or ECON 120A or MAE 108 or MATH 180A or MATH 183 or MATH 186**

- statistical inference
  - probably with closed-form null distributions (i.e. z-test, t-test, etc)

So... how does DSC 152 (this class) fit in with all this?

# What is this course about?

The focus will be on statistical data analysis and inference, leveraging what you already know from past courses. Specifically:

## What do we even mean by statistical inference?

- Hypothesis testing
  - p-values
- Confidence intervals

## And we'll use the stuff you already know and build on that

- You've learned a lot about regression in previous classes (i.e. DSC 40A, but also any AI or ML course you've taken)
- But, likely without much emphasis on doing statistical inference in the regression setting
  - In a nutshell, this will be our focus: how do we properly do statistical inference in a variety of regression settings?

# Let's go back to basics and build up from there

Example: flipping a coin

# Let's go back to basics and build up from there

To do a hypothesis test, we need to lay out our null and alternative hypotheses:

# Null distribution

In DSC 10 and DSC 80, we learned about *simulated* null distributions:

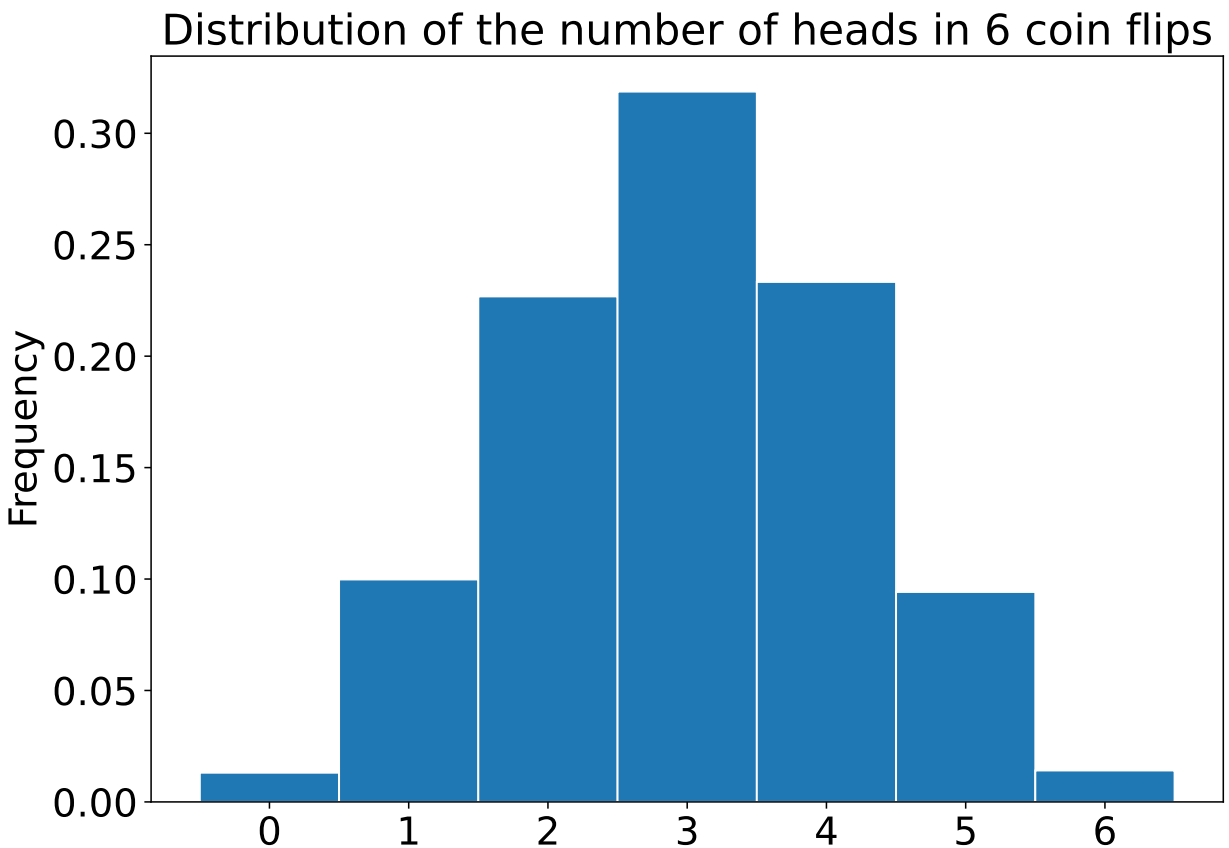
```
heads_array = np.array([])

for i in np.arange(10000):

    # Flip fair coin 6 times and count the number of Heads
    num_heads = np.random.multinomial(6, [0.5, 0.5])[0]

    # Add the number of heads seen to heads_array.
    heads_array = np.append(heads_array, num_heads)

heads_array
```

```
## array([2., 4., 3., ..., 4., 3., 2.], shape=(10000,))
```

# Null distribution

In DSC 10 and DSC 80, we learned about *simulated* null distributions:

```
(pd.DataFrame().assign(num_heads=heads_array).plot(kind='hist',
  density=True, bins=np.arange(-0.5, 6.6, 1), ec='w', legend=False,
  title = 'Distribution of the number of heads in 6 coin flips')
);
```



# Null distribution

First new thing: in this class, we are going to be using R exclusively.

## Wait what? Why?

- As data scientists, it is useful to know both Python and R

# Null distribution

First new thing: in this class, we are going to be using R exclusively.

## Wait what? Why?

- As data scientists, it is useful to know both Python and R
- Specifically, for more statistical tasks within data science, R is often a more convenient choice

# Null distribution

First new thing: in this class, we are going to be using R exclusively.

## Wait what? Why?

- As data scientists, it is useful to know both Python and R
- Specifically, for more statistical tasks within data science, R is often a more convenient choice
- Even if, in your eventual career as a data scientist, you don't need to code in R yourself, you will very possibly be working on a team where someone else does. You at least need to be able to read their code (or know when AI translates it incorrectly)

# Null distribution

First new thing: in this class, we are going to be using R exclusively.

## Wait what? Why?

- As data scientists, it is useful to know both Python and R
- Specifically, for more statistical tasks within data science, R is often a more convenient choice
- Even if, in your eventual career as a data scientist, you don't need to code in R yourself, you will very possibly be working on a team where someone else does. You at least need to be able to read their code (or know when AI translates it incorrectly)

I will continue to show you the Python equivalent here and there to help you understand what we're trying to do in R, but all of your assignments must be done in R.

# Null distribution

First new thing: in this class, we are going to be using R exclusively.

## Wait what? Why?

- As data scientists, it is useful to know both Python and R
- Specifically, for more statistical tasks within data science, R is often a more convenient choice
- Even if, in your eventual career as a data scientist, you don't need to code in R yourself, you will very possibly be working on a team where someone else does. You at least need to be able to read their code (or know when AI translates it incorrectly)

I will continue to show you the Python equivalent here and there to help you understand what we're trying to do in R, but all of your assignments must be done in R.

Lab 1 and the first discussion section will get you up to speed on using R, and R Markdown. But let's start looking at some R code now.

# Null distribution

The R equivalent of the Python code from a few slides ago:

```r
heads_array <- NULL

for(i in 1:10000){
    # Flip fair coin 6 times and count the number of Heads
    num_heads <- rbinom(n=1, size=6, prob=0.5)

    # Add the number of heads seen to heads_array.
    heads_array <- c(heads_array, num_heads)
}
```

```r
length(heads_array) # make sure the result has 10,000 elements
```

```
## [1] 10000
```

```r
heads_array[1:10] # show the first 10 iterations
```

```
##  [1] 3 5 4 3 3 3 4 2 4 4
```

# Null distribution

Actually, while that works, we don't even need to write a loop to do this:

```
heads_array <- rbinom(n=10000, size=6, prob=0.5)
```

```
length(heads_array) # make sure the result has 10,000 elements
```

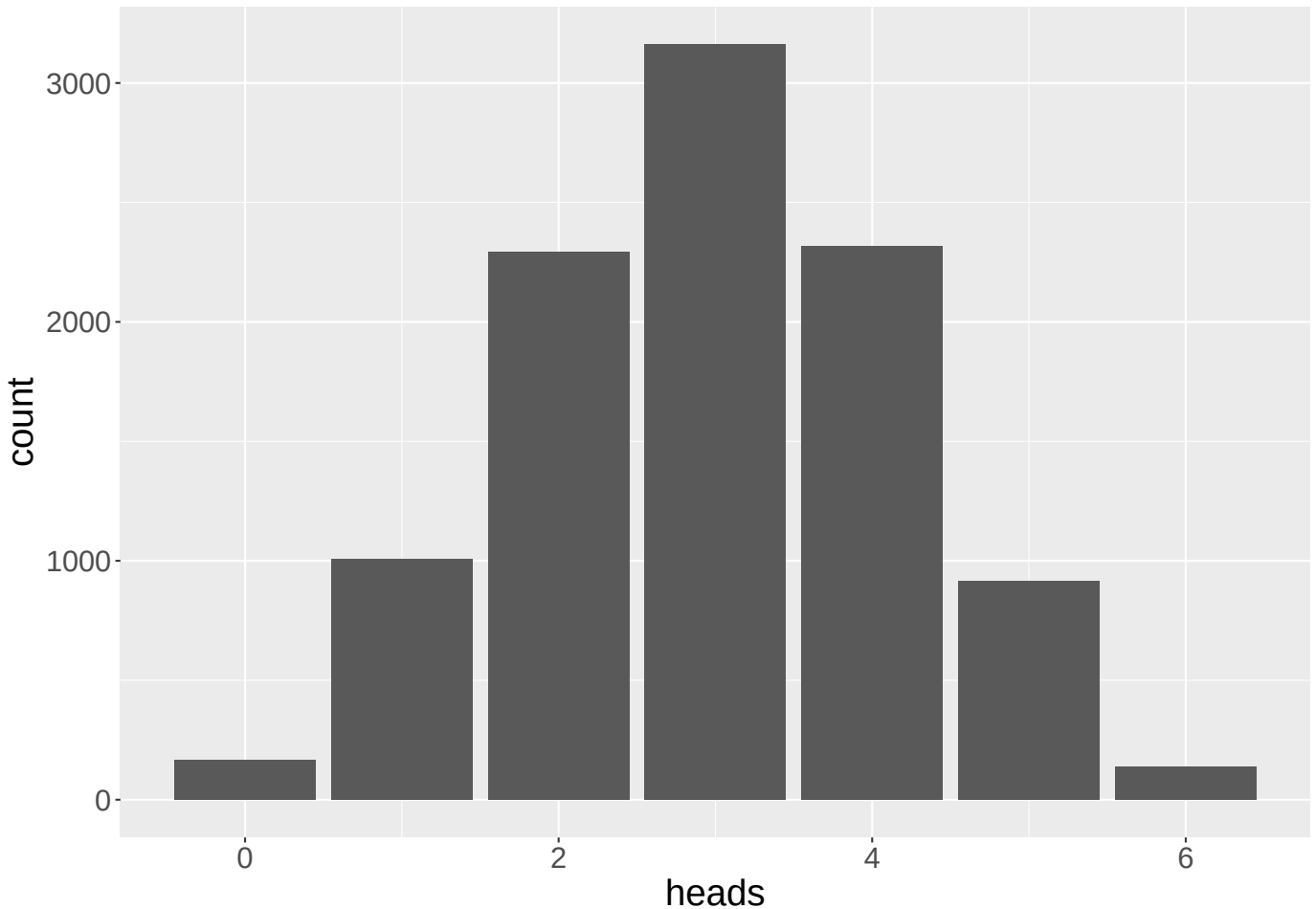
```
## [1] 10000
```

```
heads_array[1:10] # show the first 10 iterations
```

```
##  [1] 2 2 2 4 2 4 2 5 4 1
```

# Null distribution

```
heads_df <- data.frame(heads = heads_array)
ggplot(data = heads_df, aes(x=heads)) +
  geom_bar() + theme(text = element_text(size = 20))
```



# Now what?

## What is a p-value?

https://pollev.com/chi

# Now what?

**What is a p-value?**

https://pollev.com/chi

So then, what is the p-value here?

**What is the p-value here?**

https://pollev.com/chi again

# Now what?

## What is a p-value?

https://pollev.com/chi

So then, what is the p-value here?

We can also do it this way:

```
p_val <- sum(heads_array == 6 | heads_array == 0) / 10000
p_val
```

```
## [1] 0.0305
```

Note that this is a *simulated* p-value, which can only be an approximation of the theoretical p-value. The answer to the previous PollEverywhere question was the theoretical p-value!

# Binomial distribution

Flipping a coin and observing the number of heads is an example of the **Binomial Distribution** (which you sort of saw in DSC 40A, but definitely in MATH 180A, MATH 183, etc):

## Probability Mass Function of the Binomial Distribution

$$P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$$

# Binomial distribution

Flipping a coin and observing the number of heads is an example of the **Binomial Distribution** (which you sort of saw in DSC 40A, but definitely in MATH 180A, MATH 183, etc):

**Probability Mass Function of the Binomial Distribution**

$$P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$$

So,

$$P(X = 0) = \binom{6}{0} 0.5^0 (1-0.5)^{6-0} = 0.5^6 = 0.015625$$

$$P(X = 6) = \binom{6}{6} 0.5^6 (1-0.5)^{6-6} = 0.5^6 = 0.015625$$

# Theoretical p-value

$$P(X = 0) = \binom{6}{0} 0.5^0 (1 - 0.5)^{6-0} = 0.5^6 = 0.015625$$

$$P(X = 6) = \binom{6}{6} 0.5^6 (1 - 0.5)^{6-6} = 0.5^6 = 0.015625$$

Or in R:

```
dbinom(x=0, size=6, prob=0.5)
```

```
## [1] 0.015625
```

```
dbinom(x=6, size=6, prob=0.5)
```

```
## [1] 0.015625
```

# Theoretical p-value

## and so the theoretical p-value is:

```
exact_p_val <- dbinom(x=0, size=6, prob=0.5) +
               dbinom(x=6, size=6, prob=0.5)
exact_p_val
```

```
## [1] 0.03125
```

## compared to the simulated p-value:

```
p_val
```

```
## [1] 0.0305
```

So either way, we find that there is a fairly low probability of observing something at least as extreme as we did, if $H_0$ is true.

# What's the point?

## Simulated p-values as you learned in DSC 10 and 80 are good, but:

- They are more computationally expensive (not a big burden in this example, but in more complex cases it can matter)
- They (may) have lower *statistical power* than theoretical p-values (though this is often fairly minor in practice)
  - (what do we mean by statistical power? we'll talk in lots more depth about that)

## Advantages of simulated p-values

- They require fewer distributional conditions on your data (and often none whatsoever)
- They may have better Type I Error rates, if the distributional conditions of a theoretical test are not met
- They do not require you to know much math (only coding)

Both have their uses in practice, and in this class, we will use both!

# Type I Errors

Reminder: what's a Type I Error? https://pollev.com/chi

# Type I Errors

Reminder: what's a Type I Error? https://pollev.com/chi

## Rejection Region

To determine the Type I Error of a test, we first must define our rejection region.

That is, the values that are *extreme enough* to lead us to reject $H_0$.

- Suppose in our present example, we use a rejection region of $X \in \{0, 1, 5, 6\}$.
- This means that if we observe 0, 1, 5 or 6 heads out of 6 flips, we will reject $H_0$ and conclude that there is significant evidence against the coin being fair. *Does this seem reasonable??*
  - Specifically, what is the probability that we incorrectly reject $H_0$ with this rejection region?

# Type I Errors

Then, the probability of a Type I Error here would be the probability that we flip 0, 1, 5 or 6 heads *if the coin is actually fair*.

# Type I Errors

Then, the probability of a Type I Error here would be the probability that we flip 0, 1, 5 or 6 heads *if the coin is actually fair*.

What is this equal to?

# Type I Errors

Then, the probability of a Type I Error here would be the probability that we flip 0, 1, 5 or 6 heads *if the coin is actually fair*.

What is this equal to?

P(X=0) + P(X=1) + P(X=5) + P(X=6) with a fair coin...

```
TypeI <- dbinom(x=0, size=6, prob=0.5) + dbinom(x=1, size=6, prob=0.5) +
  dbinom(x=5, size=6, prob=0.5) + dbinom(x=6, size=6, prob=0.5)
TypeI
```

```
## [1] 0.21875
```

# Type I Errors

Then, the probability of a Type I Error here would be the probability that we flip 0, 1, 5 or 6 heads *if the coin is actually fair*.

What is this equal to?

P(X=0) + P(X=1) + P(X=5) + P(X=6) with a fair coin...

```
TypeI <- dbinom(x=0, size=6, prob=0.5) + dbinom(x=1, size=6, prob=0.5) +
  dbinom(x=5, size=6, prob=0.5) + dbinom(x=6, size=6, prob=0.5)
TypeI
```

```
## [1] 0.21875
```

So if we had decided to use $\{0, 1, 5, 6\}$ as our rejection region, then there would be a 21.875% chance of making a Type I Error.

... that's pretty high!

# Type I Errors

**$\alpha$-level test**

A statistical test with an $\alpha$ probability of making a Type I Error is referred to as an $\alpha$-level test.

So the example on the previous slide would be a 0.21875-level test.

The most commonly used value for $\alpha$ is 0.05. So in that case, we would be doing a 0.05-level test.

But often, $\alpha$ is only an approximation, and/or relies on distributional conditions to be correct!

Next time: what exactly do we mean by that?

# To-dos

- If you do not already have R and RStudio installed on your machine, get them installed!
  - https://posit.co/download/rstudio-desktop/

- Tomorrow's discussion section will be an introduction to R and R Markdown. If you are new to either of these, please plan to attend.
  - You may attend either the 3pm or the 4pm section (both are in Center Hall 216).

- Read the course syllabus and complete the Syllabus Check by tomorrow 4/1 at midnight

- Complete the Welcome Survey by tomorrow 4/1 at midnight

- Complete today's Daily Check by **tonight at midnight**