

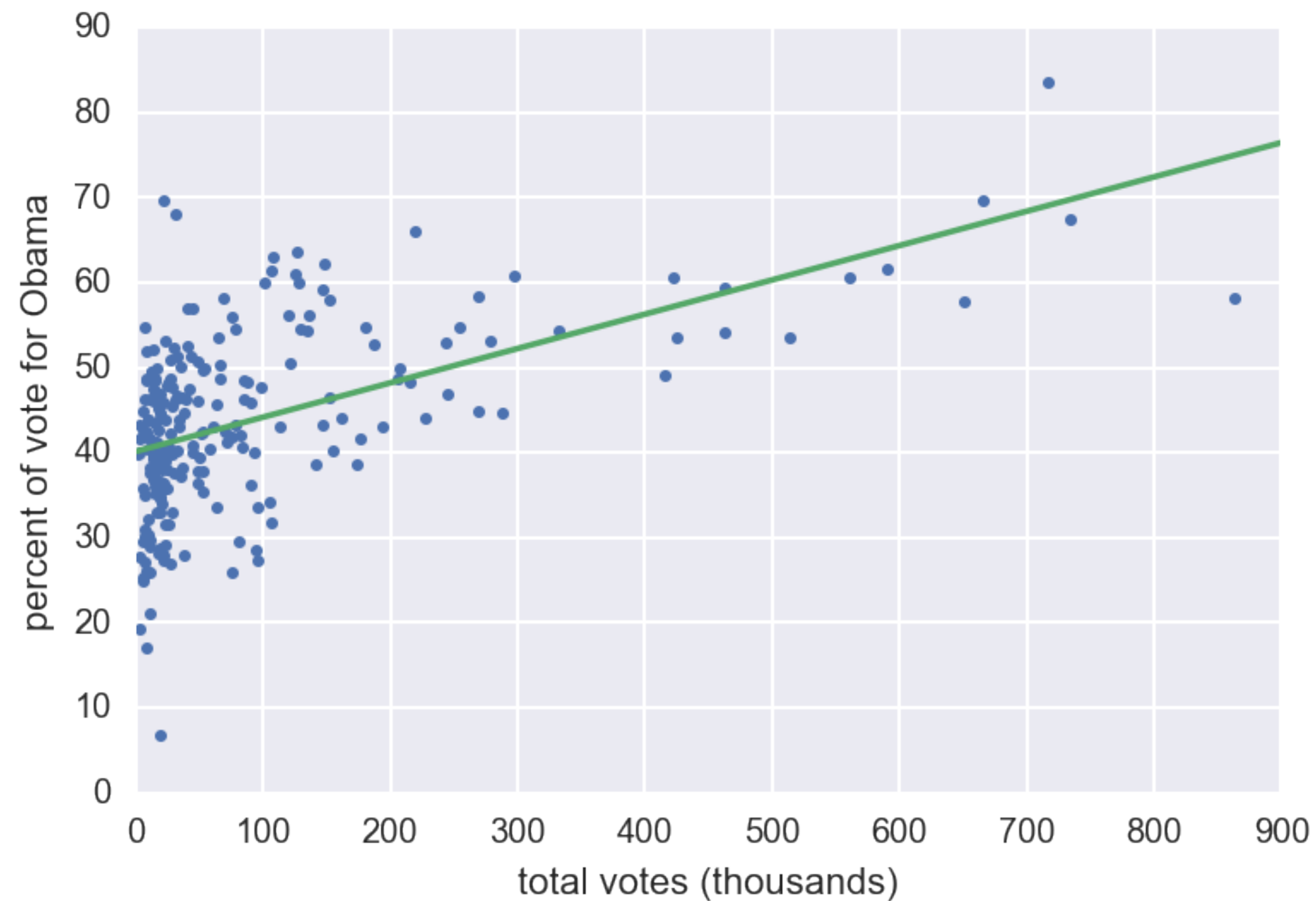


STATISTICAL THINKING IN PYTHON II

Formulating and simulating hypotheses



2008 US swing state election results





Hypothesis testing

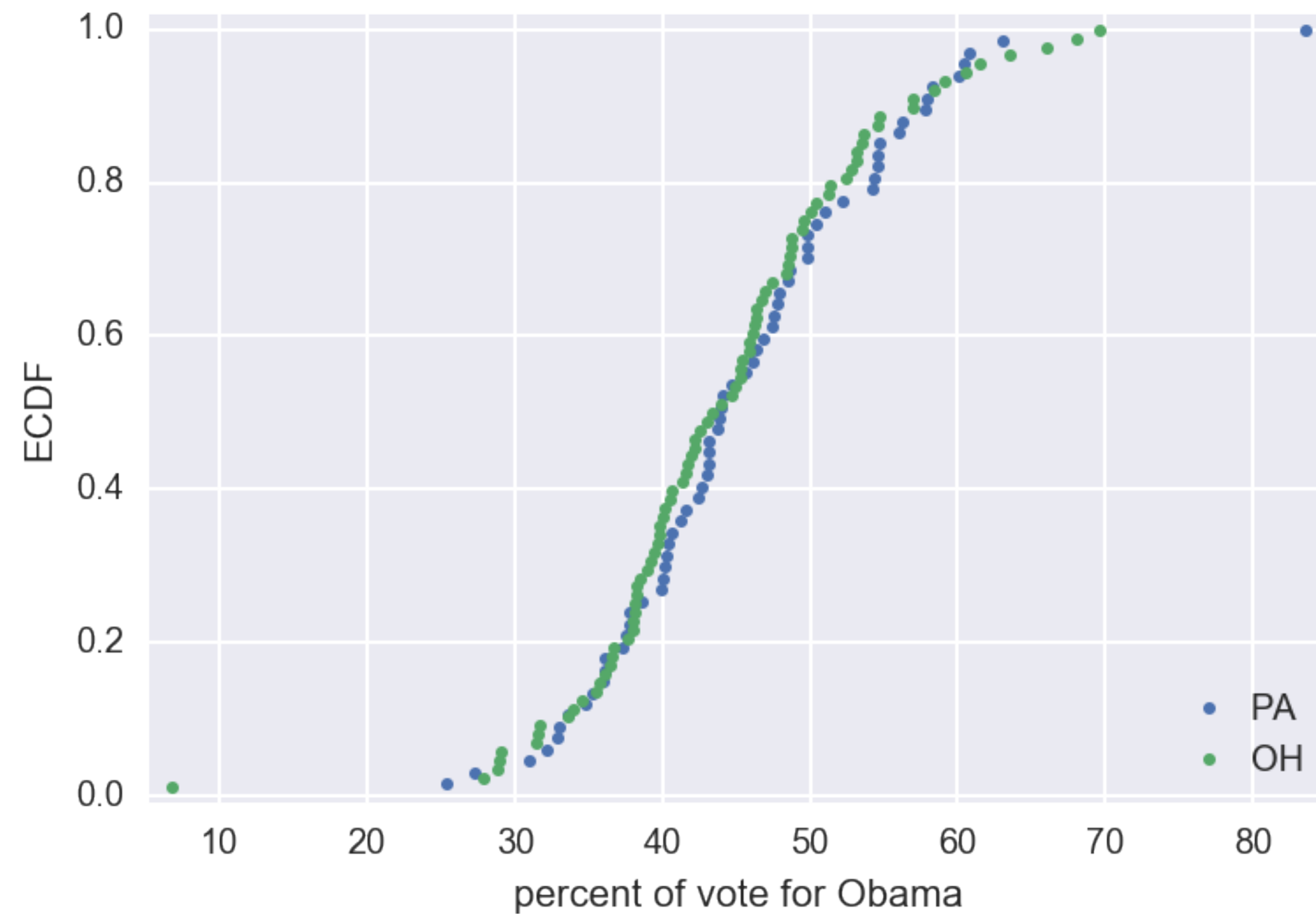
- Assessment of how reasonable the observed data are assuming a hypothesis is true

Null hypothesis

- Another name for the hypothesis you are testing



ECDFs of swing state election results





Percent vote for Obama

	PA	OH	PA — OH difference
mean	45.5%	44.3%	1.2%
median	44.0%	43.7%	0.4%
standard deviation	9.8%	9.9%	—0.1%



Simulating the hypothesis

60.08,	40.64,	36.07,	41.21,	31.04,	43.78,	44.08,	46.85,
44.71,	46.15,	63.10,	52.20,	43.18,	40.24,	39.92,	47.87,
37.77,	40.11,	49.85,	48.61,	38.62,	54.25,	34.84,	47.75,
43.82,	55.97,	58.23,	42.97,	42.38,	36.11,	37.53,	42.65,
50.96,	47.43,	56.24,	45.60,	46.39,	35.22,	48.56,	32.97,
57.88,	36.05,	37.72,	50.36,	32.12,	41.55,	54.66,	57.81,
54.58,	32.88,	54.37,	40.45,	47.61,	60.49,	43.11,	27.32,
44.03,	33.56,	37.26,	54.64,	43.12,	25.34,	49.79,	83.56,
40.09,	60.81,	49.81,	56.94,	50.46,	65.99,	45.88,	42.23,
45.26,	57.01,	53.61,	59.10,	61.48,	43.43,	44.69,	54.59,
48.36,	45.89,	48.62,	43.92,	38.23,	28.79,	63.57,	38.07,
40.18,	43.05,	41.56,	42.49,	36.06,	52.76,	46.07,	39.43,
39.26,	47.47,	27.92,	38.01,	45.45,	29.07,	28.94,	51.28,
50.10,	39.84,	36.43,	35.71,	31.47,	47.01,	40.10,	48.76,
31.56,	39.86,	45.31,	35.47,	51.38,	46.33,	48.73,	41.77,
41.32,	48.46,	53.14,	34.01,	54.74,	40.67,	38.96,	46.29,
38.25,	6.80,	31.75,	46.33,	44.90,	33.57,	38.10,	39.67,
40.47,	49.44,	37.62,	36.71,	46.73,	42.20,	53.16,	52.40,
58.36,	68.02,	38.53,	34.58,	69.64,	60.50,	53.53,	36.54,
49.58,	41.97,	38.11,					

Pennsylvania

Ohio



Simulating the hypothesis

```
60.08, 40.64, 36.07, 41.21, 31.04, 43.78, 44.08, 46.85,  
44.71, 46.15, 63.10, 52.20, 43.18, 40.24, 39.92, 47.87,  
37.77, 40.11, 49.85, 48.61, 38.62, 54.25, 34.84, 47.75,  
43.82, 55.97, 58.23, 42.97, 42.38, 36.11, 37.53, 42.65,  
50.96, 47.43, 56.24, 45.60, 46.39, 35.22, 48.56, 32.97,  
57.88, 36.05, 37.72, 50.36, 32.12, 41.55, 54.66, 57.81,  
54.58, 32.88, 54.37, 40.45, 47.61, 60.49, 43.11, 27.32,  
44.03, 33.56, 37.26, 54.64, 43.12, 25.34, 49.79, 83.56,  
40.09, 60.81, 49.81, 56.94, 50.46, 65.99, 45.88, 42.23,  
45.26, 57.01, 53.61, 59.10, 61.48, 43.43, 44.69, 54.59,  
48.36, 45.89, 48.62, 43.92, 38.23, 28.79, 63.57, 38.07,  
40.18, 43.05, 41.56, 42.49, 36.06, 52.76, 46.07, 39.43,  
39.26, 47.47, 27.92, 38.01, 45.45, 29.07, 28.94, 51.28,  
50.10, 39.84, 36.43, 35.71, 31.47, 47.01, 40.10, 48.76,  
31.56, 39.86, 45.31, 35.47, 51.38, 46.33, 48.73, 41.77,  
41.32, 48.46, 53.14, 34.01, 54.74, 40.67, 38.96, 46.29,  
38.25, 6.80, 31.75, 46.33, 44.90, 33.57, 38.10, 39.67,  
40.47, 49.44, 37.62, 36.71, 46.73, 42.20, 53.16, 52.40,  
58.36, 68.02, 38.53, 34.58, 69.64, 60.50, 53.53, 36.54,  
49.58, 41.97, 38.11
```



Simulating the hypothesis

```
59.10, 38.62, 51.38, 60.49, 6.80, 41.97, 48.56, 37.77,  
48.36, 54.59, 40.11, 57.81, 45.89, 83.56, 40.64, 46.07,  
28.79, 55.97, 33.57, 42.23, 48.61, 44.69, 39.67, 57.88,  
48.62, 54.66, 54.74, 48.46, 36.07, 43.92, 49.85, 53.53,  
48.76, 41.77, 36.54, 47.01, 52.76, 49.44, 34.58, 40.24,  
44.08, 46.29, 49.81, 69.64, 60.50, 27.32, 45.60, 63.10,  
35.71, 39.86, 40.67, 65.99, 50.46, 37.72, 50.96, 42.49,  
31.56, 38.23, 37.26, 41.21, 37.53, 46.85, 44.03, 41.32,  
45.88, 40.45, 32.12, 35.22, 49.79, 43.12, 43.18, 45.45,  
25.34, 46.73, 44.90, 56.94, 58.23, 39.84, 36.05, 43.05,  
38.25, 40.47, 31.04, 54.25, 46.15, 57.01, 52.20, 47.75,  
36.06, 47.61, 51.28, 43.43, 42.97, 38.01, 54.64, 45.26,  
47.47, 34.84, 49.58, 48.73, 29.07, 54.58, 27.92, 34.01,  
38.07, 31.47, 36.11, 39.26, 41.56, 52.40, 40.18, 47.87,  
46.33, 46.39, 43.11, 38.53, 33.56, 42.65, 68.02, 35.47,  
40.09, 36.43, 36.71, 60.08, 50.36, 39.43, 28.94, 58.36,  
42.20, 47.43, 44.71, 43.78, 39.92, 37.62, 63.57, 53.61,  
40.10, 46.33, 53.16, 32.88, 38.96, 41.55, 56.24, 38.11,  
42.38, 38.10, 43.82, 45.31, 60.81, 54.37, 53.14, 32.97,  
61.48, 50.10, 31.75
```



Simulating the hypothesis

59.10,	38.62,	51.38,	60.49,	6.80,	41.97,	48.56,	37.77,
48.36,	54.59,	40.11,	57.81,	45.89,	83.56,	40.64,	46.07,
28.79,	55.97,	33.57,	42.23,	48.61,	44.69,	39.67,	57.88,
48.62,	54.66,	54.74,	48.46,	36.07,	43.92,	49.85,	53.53,
48.76,	41.77,	36.54,	47.01,	52.76,	49.44,	34.58,	40.24,
44.08,	46.29,	49.81,	69.64,	60.50,	27.32,	45.60,	63.10,
35.71,	39.86,	40.67,	65.99,	50.46,	37.72,	50.96,	42.49,
31.56,	38.23,	37.26,	41.21,	37.53,	46.85,	44.03,	41.32,
45.88,	40.45,	32.12,	35.22,	49.79,	43.12,	43.18,	45.45,
25.34,	46.73,	44.90,	56.94,	58.23,	39.84,	36.05,	43.05,
38.25,	40.47,	31.04,	54.25,	46.15,	57.01,	52.20,	47.75,
36.06,	47.61,	51.28,	43.43,	42.97,	38.01,	54.64,	45.26,
47.47,	34.84,	49.58,	48.73,	29.07,	54.58,	27.92,	34.01,
38.07,	31.47,	36.11,	39.26,	41.56,	52.40,	40.18,	47.87,
46.33,	46.39,	43.11,	38.53,	33.56,	42.65,	68.02,	35.47,
40.09,	36.43,	36.71,	60.08,	50.36,	39.43,	28.94,	58.36,
42.20,	47.43,	44.71,	43.78,	39.92,	37.62,	63.57,	53.61,
40.10,	46.33,	53.16,	32.88,	38.96,	41.55,	56.24,	38.11,
42.38,	38.10,	43.82,	45.31,	60.81,	54.37,	53.14,	32.97,
61.48,	50.10,	31.75,					

"Pennsylvania"

"Ohio"

Permutation

- Random reordering of entries in an array



Generating a permutation sample

```
In [1]: import numpy as np

In [2]: dem_share_both = np.concatenate(
...:     (dem_share_PA, dem_share_OH))

In [3]: dem_share_perm = np.random.permutation(dem_share_both)

In [4]: perm_sample_PA = dem_share_perm[:len(dem_share_PA)]

In [5]: perm_sample_OH = dem_share_perm[len(dem_share_PA):]
```



STATISTICAL THINKING IN PYTHON II

Let's practice!

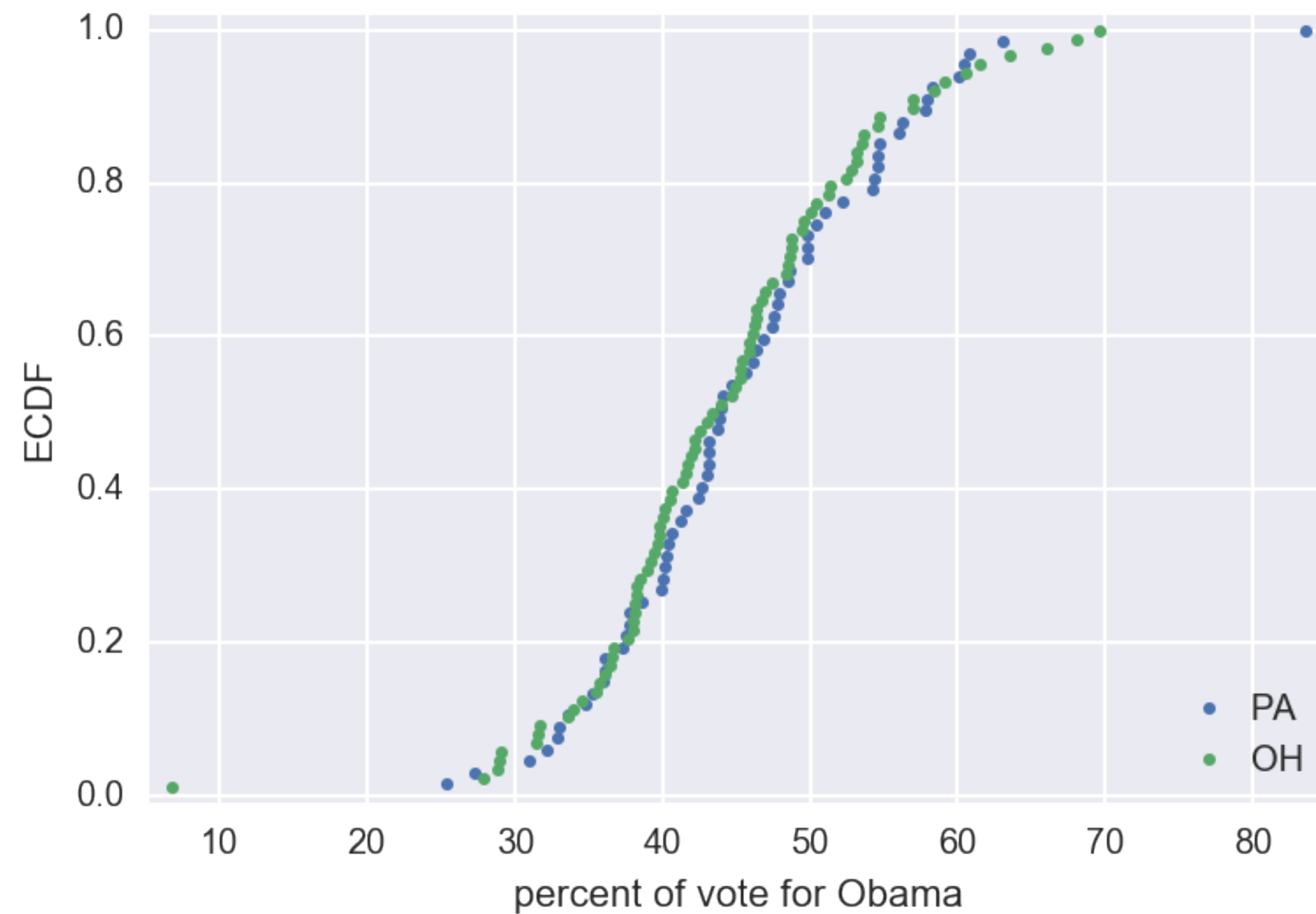


STATISTICAL THINKING IN PYTHON II

Test statistics and p-values



Are OH and PA different?



Hypothesis testing

- Assessment of how reasonable the observed data are assuming a hypothesis is true

Test statistic

- A single number that can be computed from observed data and from data you simulate under the null hypothesis
- It serves as a basis of comparison between the two



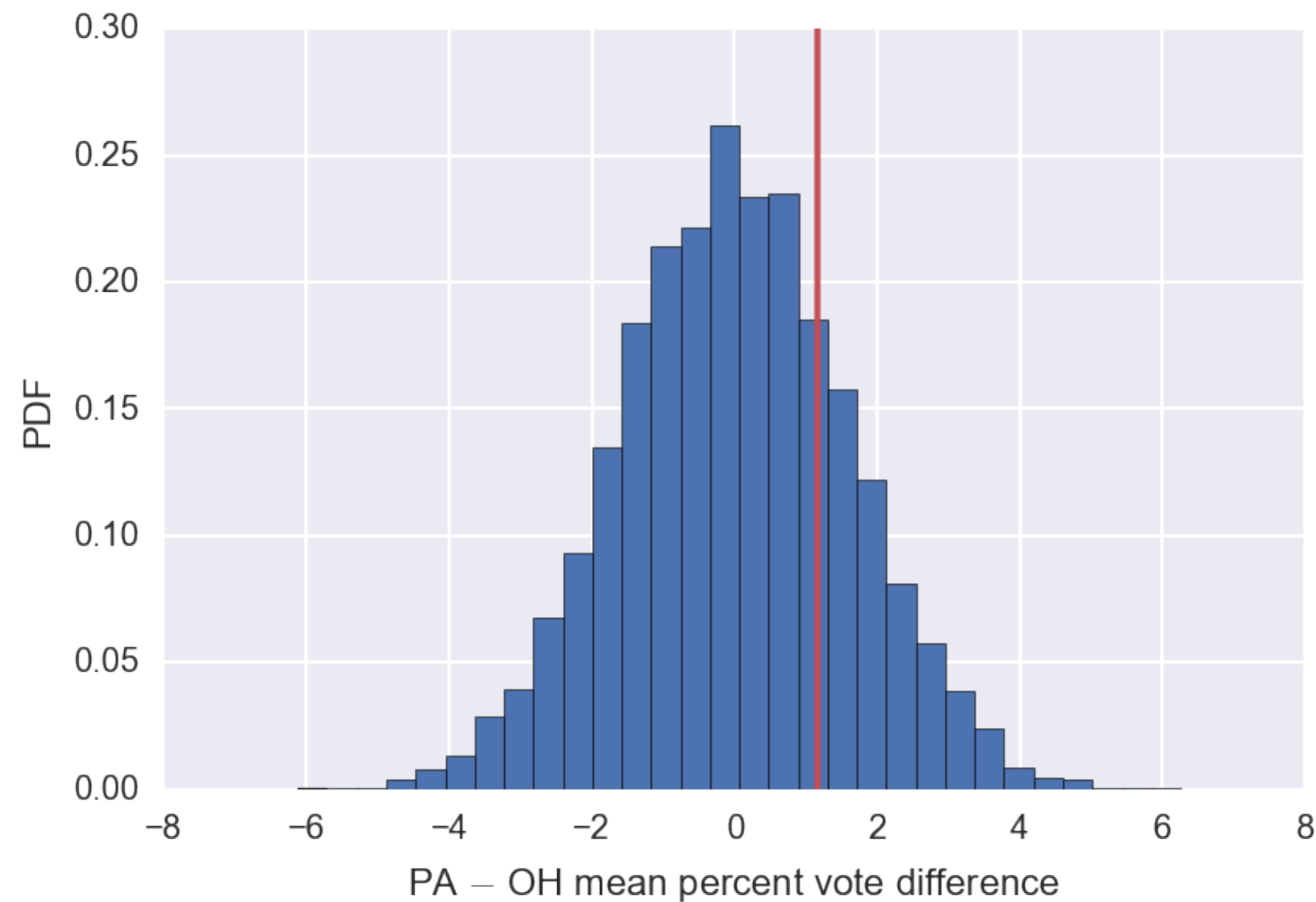
Permutation replicate

```
In [1]: np.mean(perm_sample_PA) - np.mean(perm_sample_OH)
Out[1]: 1.122220149253728
```

```
In [2]: np.mean(dem_share_PA) - np.mean(dem_share_OH) # orig. data
Out[2]: 1.1582360922659518
```

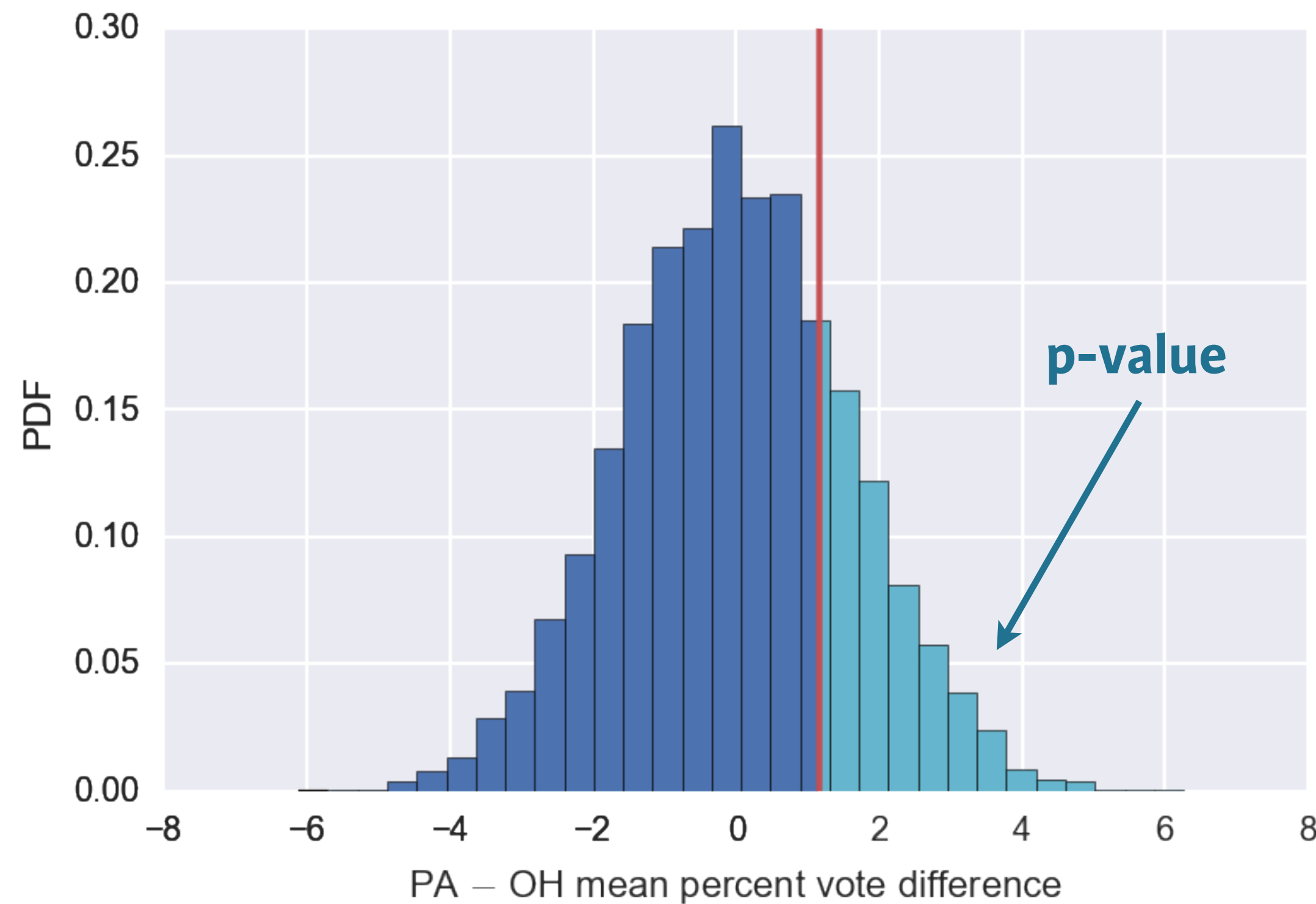


Mean vote difference under null hypothesis





Mean vote difference under null hypothesis



p-value

- The probability of obtaining a value of your test statistic that is at least as extreme as what was observed, under the assumption the null hypothesis is true
- **NOT** the probability that the null hypothesis is true

Statistical significance

- Determined by the smallness of a p-value

Null hypothesis significance testing (NHST)

- Another name for what we are doing in this chapter

statistical significance \neq practical significance



STATISTICAL THINKING IN PYTHON II

Let's practice!



STATISTICAL THINKING IN PYTHON II

Bootstrap hypothesis tests

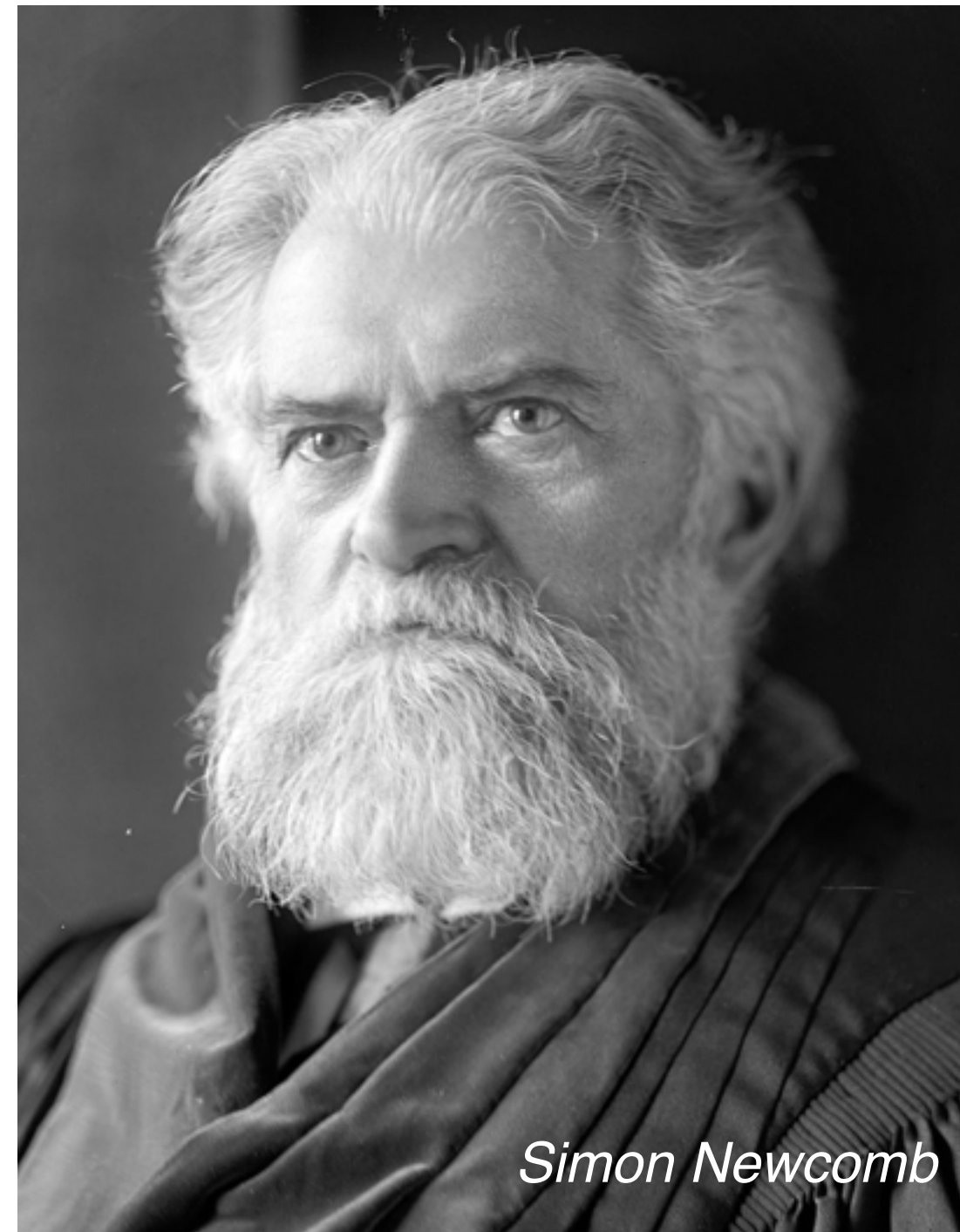
Pipeline for hypothesis testing

- Clearly state the null hypothesis
- Define your test statistic
- Generate many sets of simulated data assuming the null hypothesis is true
- Compute the test statistic for each simulated data set
- The p-value is the fraction of your simulated data sets for which the test statistic is at least as extreme as for the real data

Michelson and Newcomb: speed of light pioneers



299,852 km/s



299,860 km/s

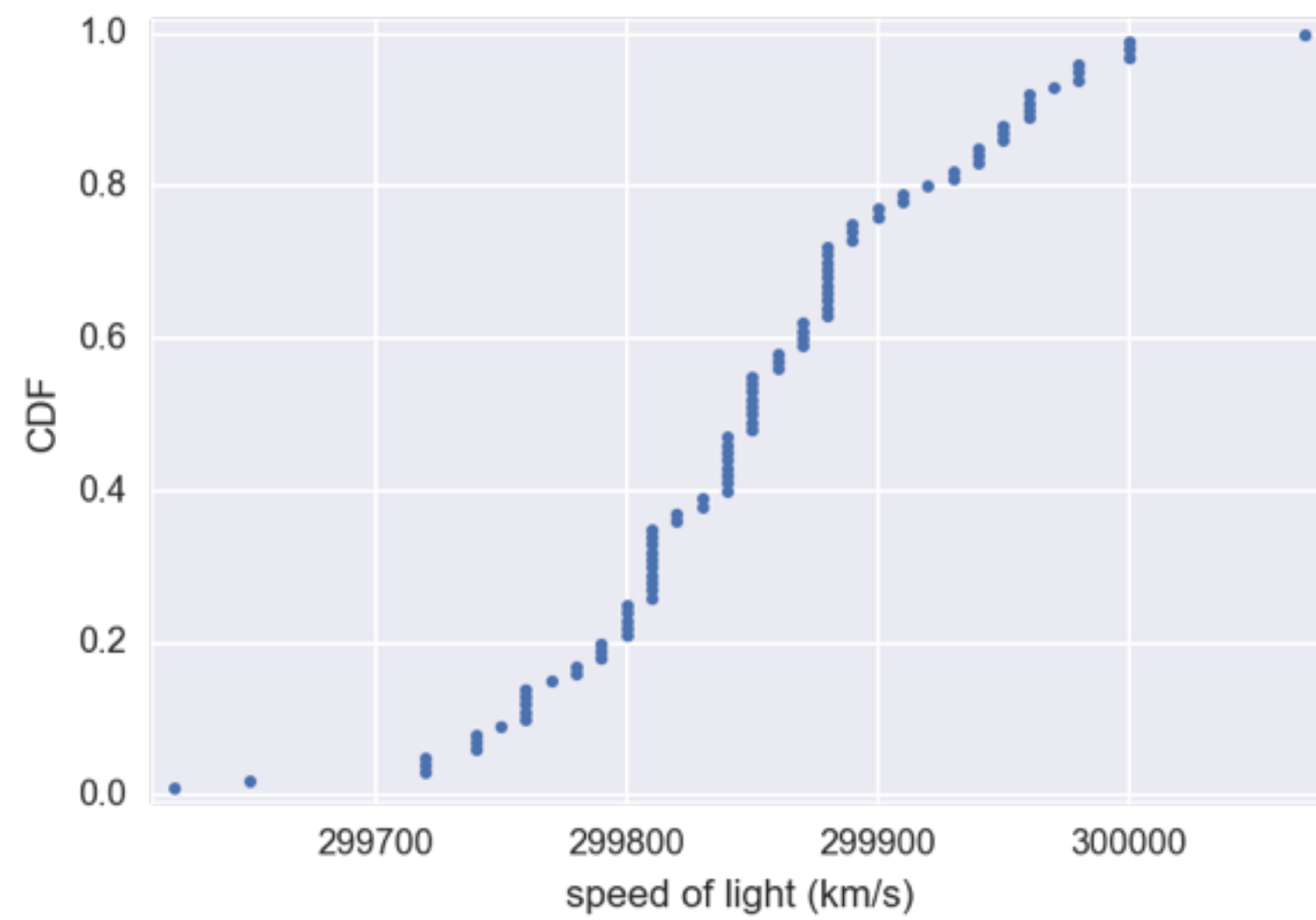
Michelson image: public domain, Smithsonian

Newcomb image: US Library of Congress

The data we have

Michelson:

Newcomb:



mean = 299,860 km/s

Null hypothesis

- The true mean speed of light in Michelson's experiments was actually Newcomb's reported value



Shifting the Michelson data

```
In [1]: newcomb_value = 299860 # km/s
```

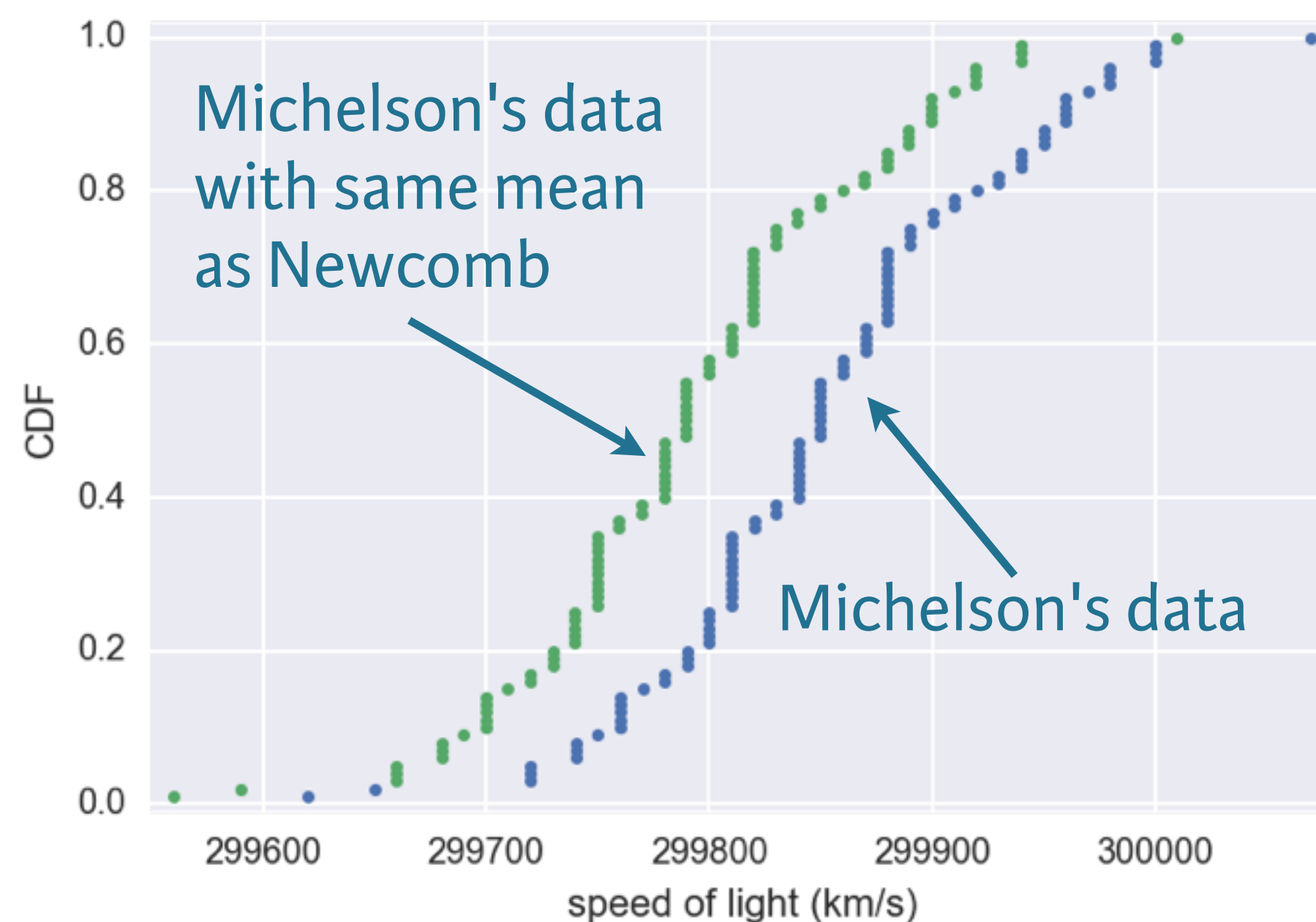
```
In [2]: michelson_shifted = michelson_speed_of_light \
...:     - np.mean(michelson_speed_of_light) + newcomb_value
```



Shifting the Michelson data

```
In [1]: newcomb_value = 299860 # km/s
```

```
In [2]: michelson_shifted = michelson_speed_of_light \
....:    - np.mean(michelson_speed_of_light) + newcomb_value
```





Calculating the test statistic

```
In [1]: def diff_from_newcomb(data, newcomb_value=299860):  
...:     return np.mean(data) - newcomb_value  
...:  
  
In [2]: diff_obs = diff_from_newcomb(michelson_speed_of_light)  
  
In [3]: diff_obs  
Out[3]: -7.59999999999767169
```



Computing the p-value

```
In [1]: bs_replicates = draw_bs_reps(michelson_shifted,  
....:                               diff_from_newcomb, 10000)  
  
In [2]: p_value = np.sum(bs_replicates <= diff_observed) / 10000  
  
In [3]: p_value  
Out[3]: 0.16039999999999999
```

One sample test

- Compare one set of data to a single number

Two sample test

- Compare two sets of data



STATISTICAL THINKING IN PYTHON II

Let's practice!