Activity of zebrafish and melatonin

CASE STUDIES IN STATISTICAL THINKING

Justin Bois
Lecturer, Caltech

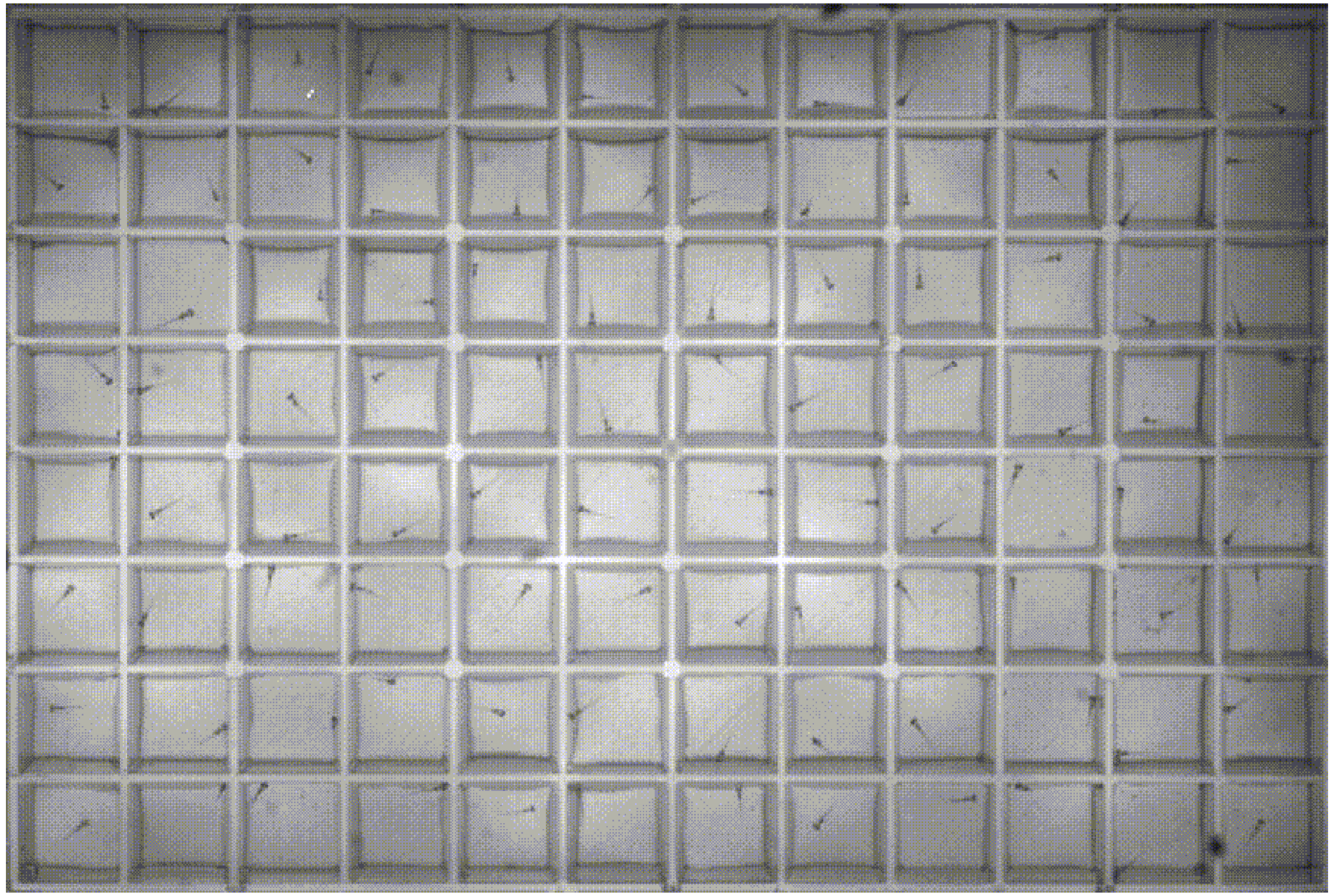
datacamp

Caltech

Case studies in statistical thinking

- Hone and extend your statistical thinking skills
- Work with real data sets
- Review of Statistical Thinking I and II

Warming up with zebrafish

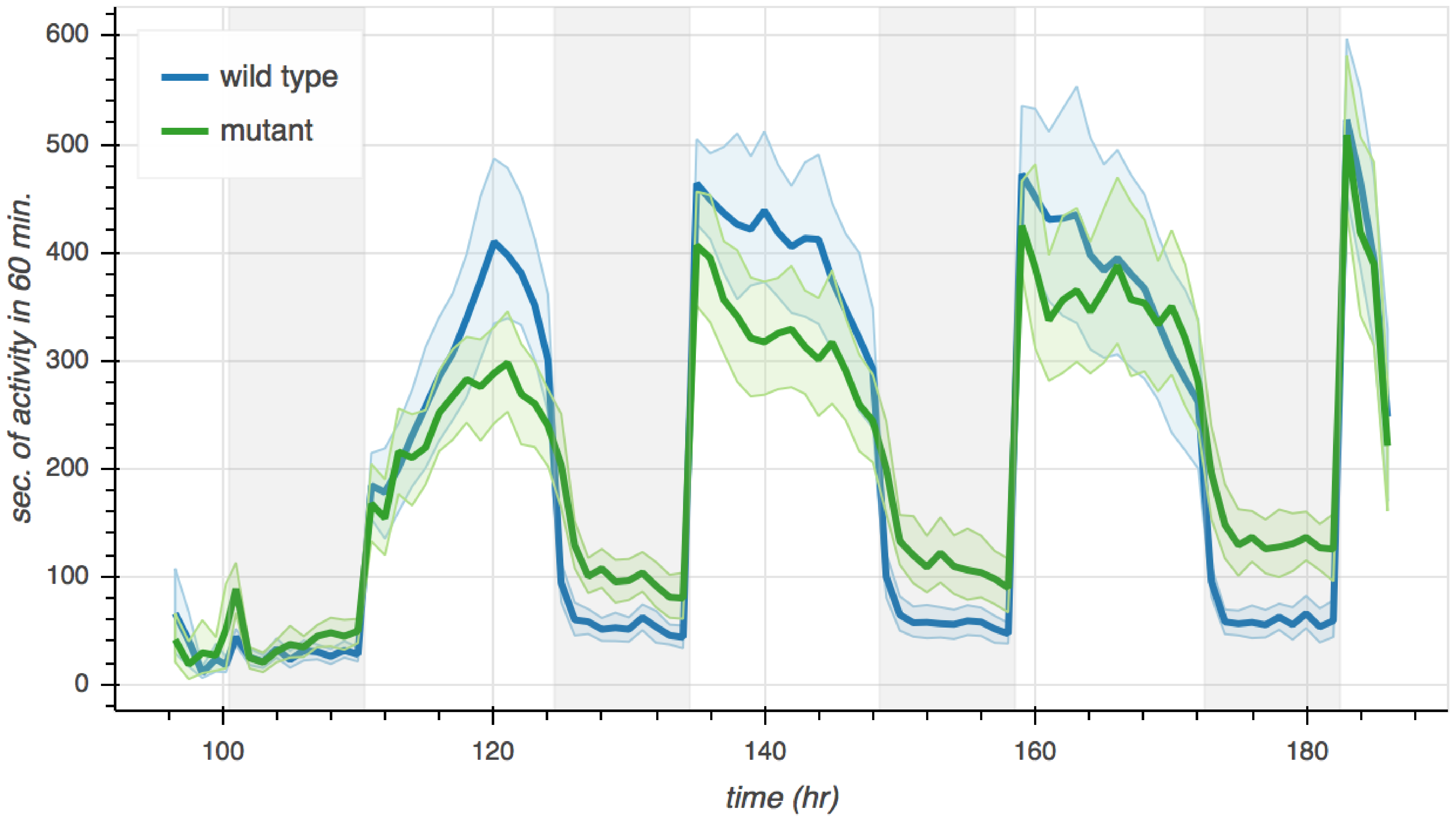


[1] Movie courtesy of David Prober, Caltech

Nomenclature

- **Mutant:** Has the mutation on both chromosomes
- **Wild type:** Does not have the mutation

Activity of fish, day and night



[1] Data courtesy of Avni Gandhi, Grigorios Oikonomou, and David Prober, Caltech

Active bouts: a metric for wakefulness

- **Active bout:** A period of time where a fish is consistently active
- **Active bout length:** Number of consecutive minutes with activity

Probability distributions and stories

- **Probability distribution**: A mathematical description of outcomes
- A probability distribution has a **story**

Distributions from Statistical Thinking I

- Uniform
- Binomial
- Poisson
- Normal
- Exponential

The Exponential distribution

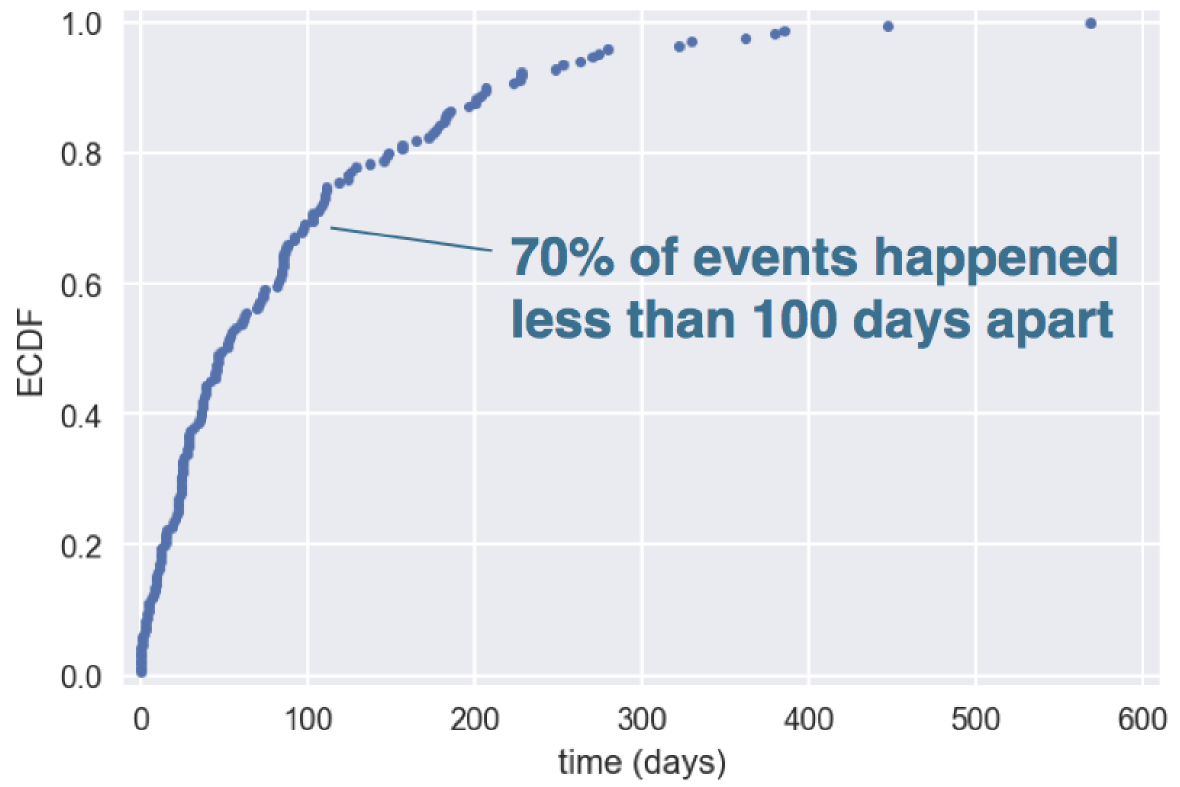
- **Poisson process:** The timing of the next event is completely independent of when the previous event happened

- **Story of the Exponential distribution:** The waiting time between arrivals of a Poisson process is Exponentially distributed

The Exponential CDF

```
x, y = ecdf(nuclear_incident_times)

_ = plt.plot(x, y, marker='.', linestyle='none')
```

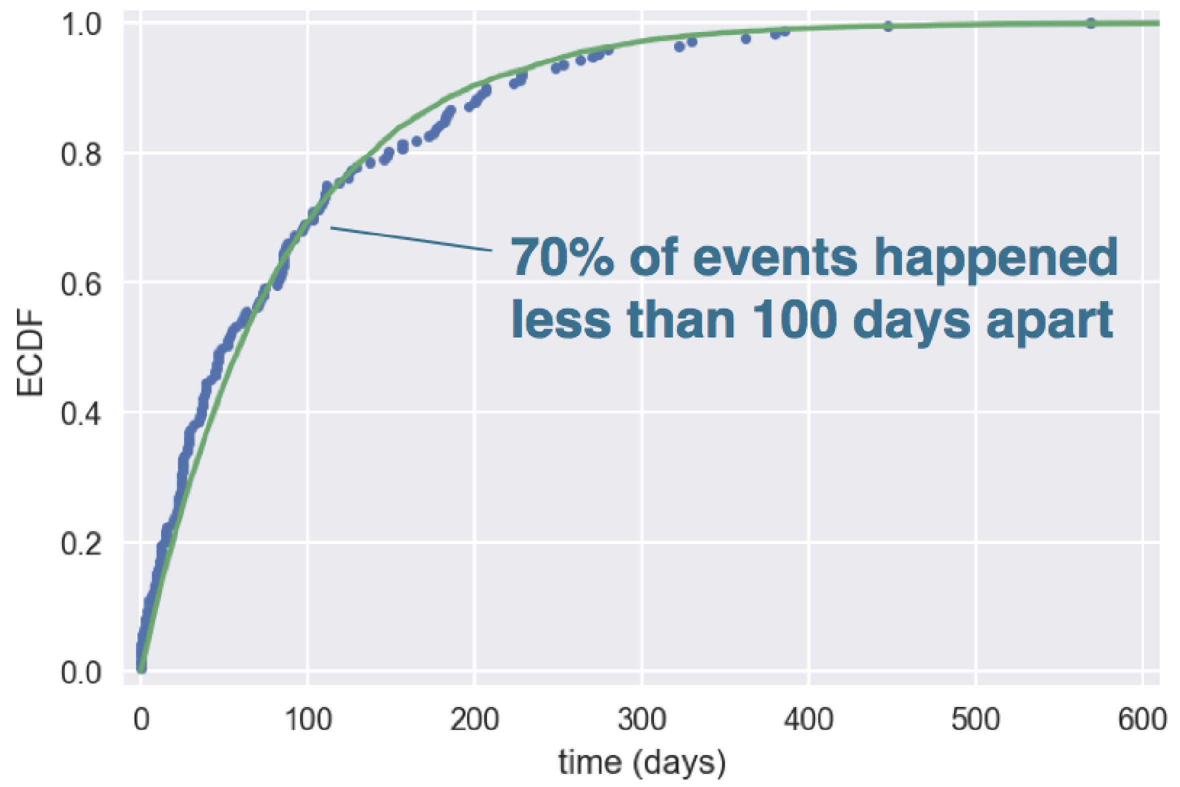


[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

The Exponential CDF

```
x, y = ecdf(nuclear_incident_times)

_ = plt.plot(x, y, marker='.', linestyle='none')
```



[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

```python
import dc_stat_think as dcst
dcst.pearson_r?
```

```
Signature: dcst.pearson_r(data_1, data_2)
Docstring: Compute the Pearson correlation coefficient between two
samples.
Parameters
----------
data_1 : array_like
    One-dimensional array of data.
data_2 : array_like
    One-dimensional array of  data.
Returns
-------
output : float
    The Pearson correlation coefficient between `data_1`
    and `data_2`.
File:      usr/local/lib/python3.5/site-packages/
           dc_stat_think-0.1.4-py3.6.egg/dc_stat_think/dc_stat_think.py
Type:      function
```

Using the dc_stat_think module

```
x, y = dcst.ecdf(nuclear_incident_times)
```

```
% pip install dc_stat_think
```

Let's practice!

CASE STUDIES IN STATISTICAL THINKING

Bootstrap confidence intervals

CASE STUDIES IN STATISTICAL THINKING

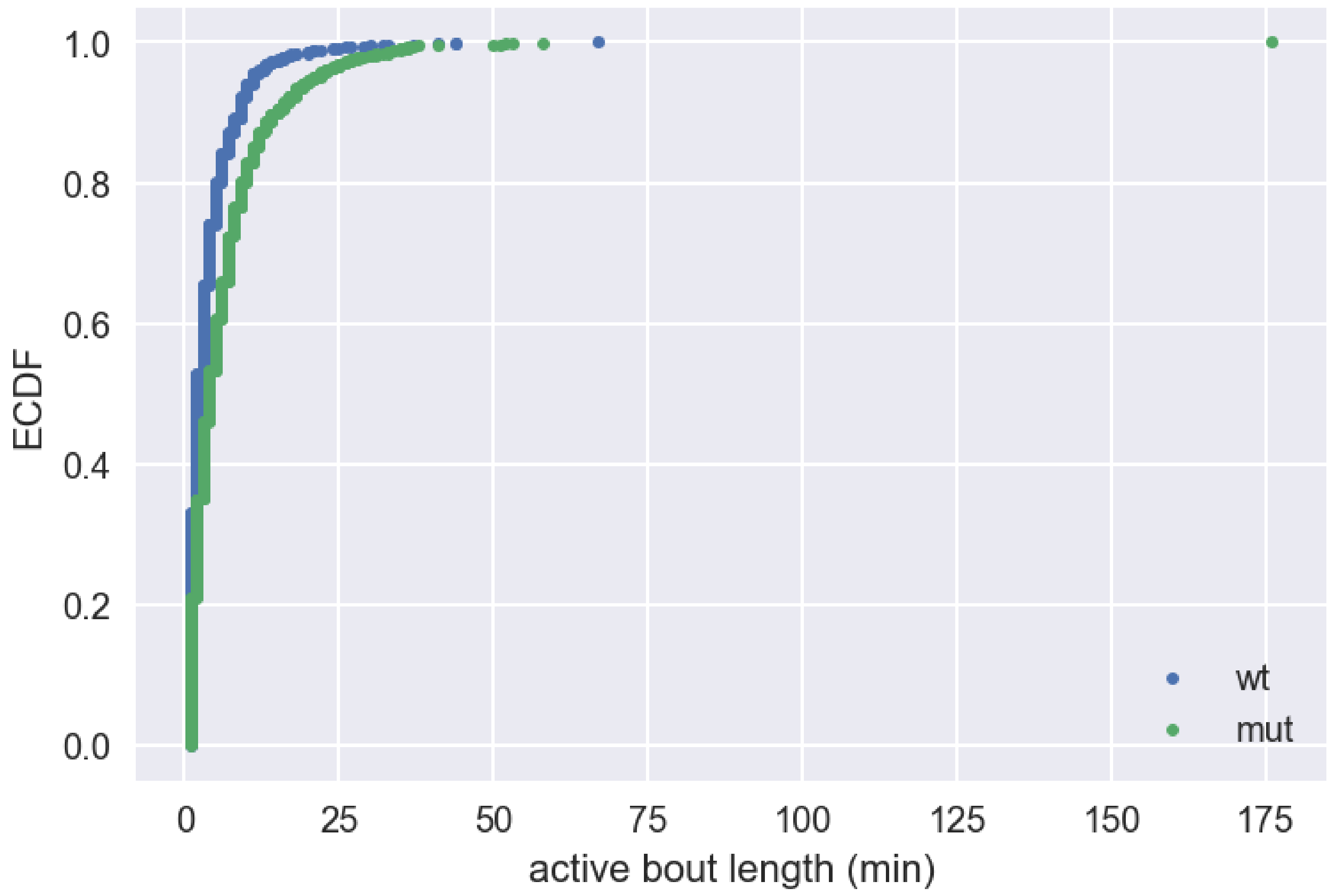
Justin Bois
Lecturer, Caltech

EDA is the first step

"Exploratory data analysis can never be the whole story, but nothing else can serve as a foundation stone, as the first step."

--John Tukey

Active bout length ECDFs



[1] Data courtesy of Avni Gandhi, Grigorios Oikonomou, and David Prober, Caltech

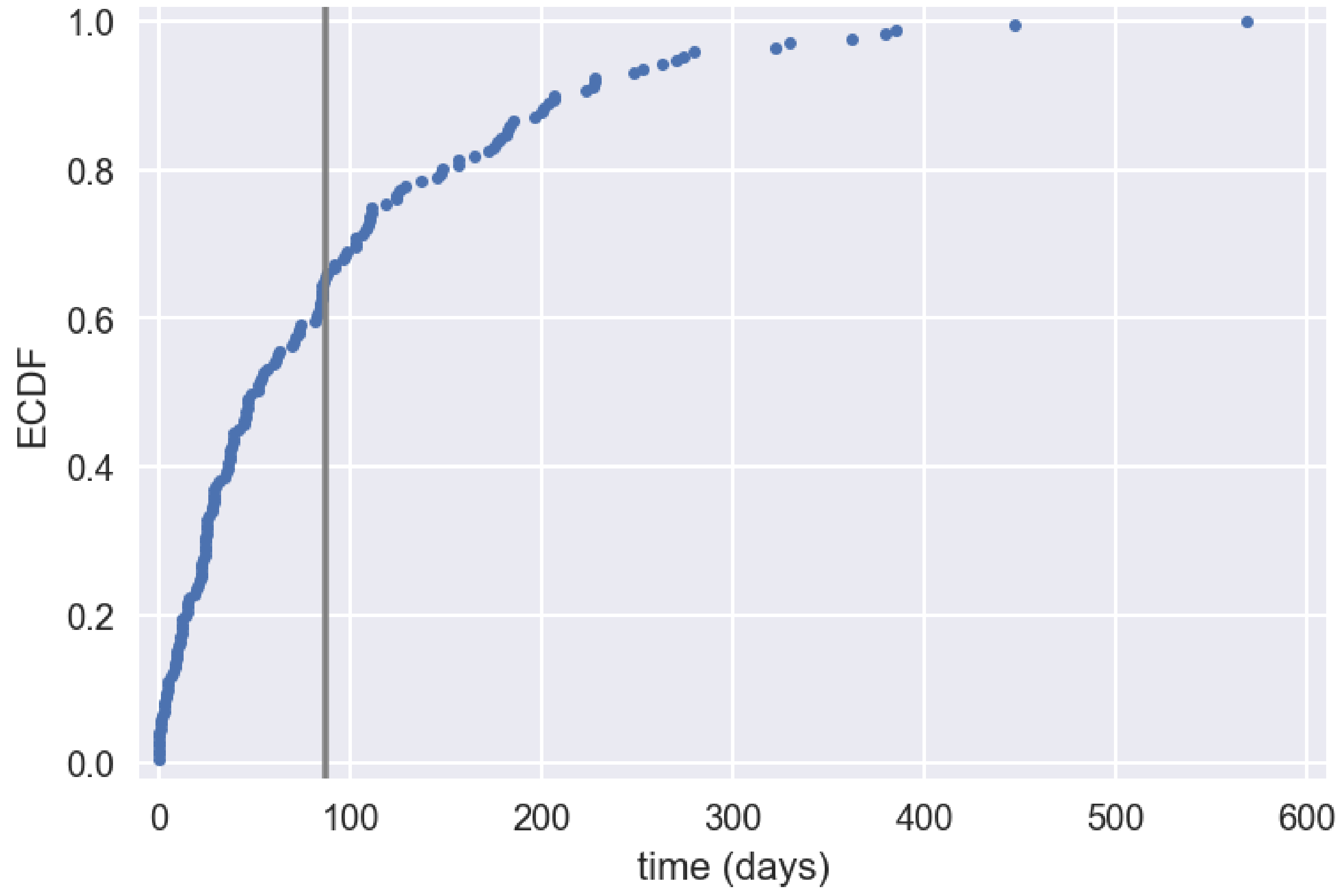
Optimal parameter value

- **Optimal parameter value:** The value of the parameter of a probability distribution that best describes the data

- **Optimal parameter for the Exponential distribution:** Computed from the mean of the data

```
np.mean(nuclear_incident_times)
```

```
87.14035087719298 6
```



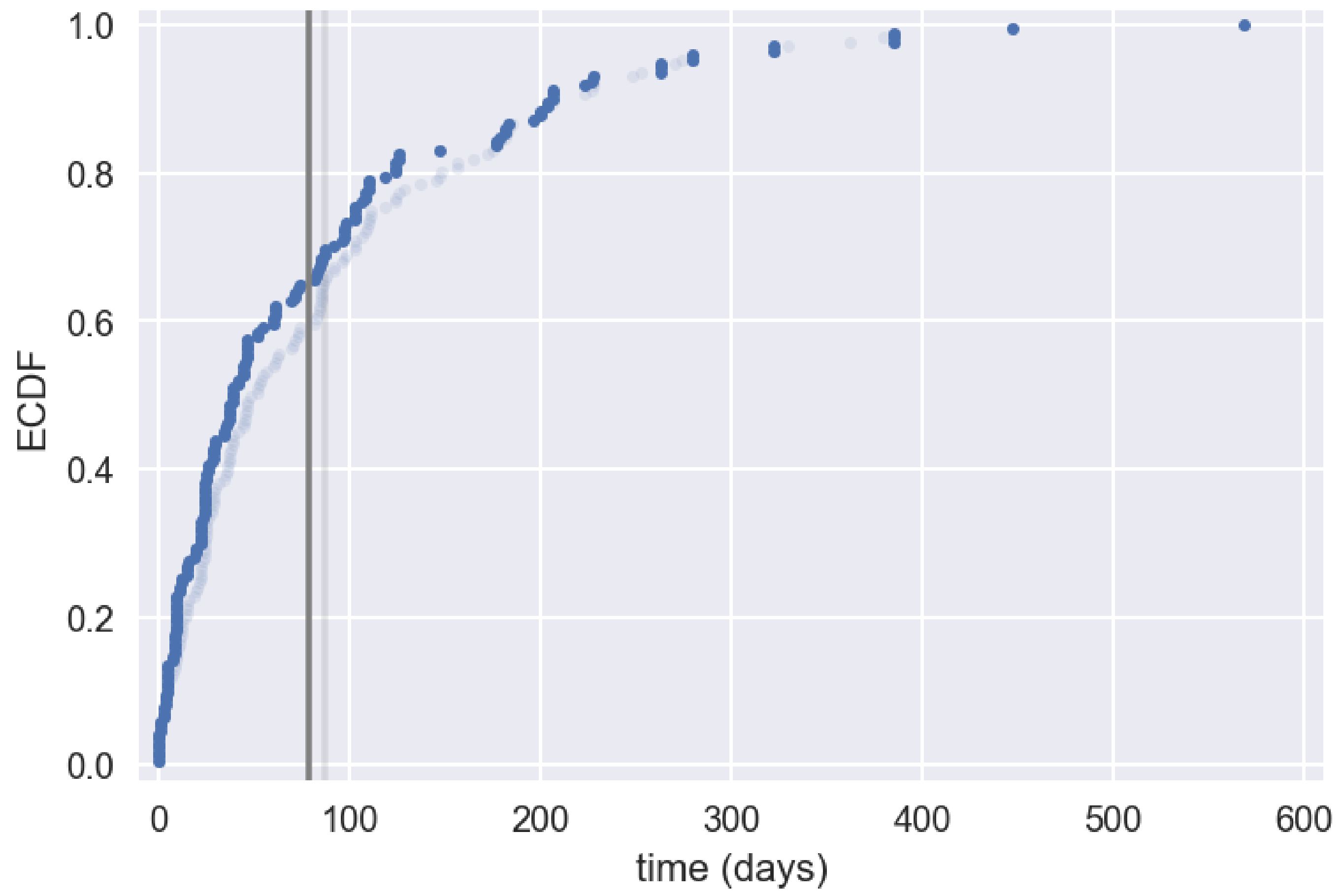
[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

Bootstrap sample

A resampled array of the data

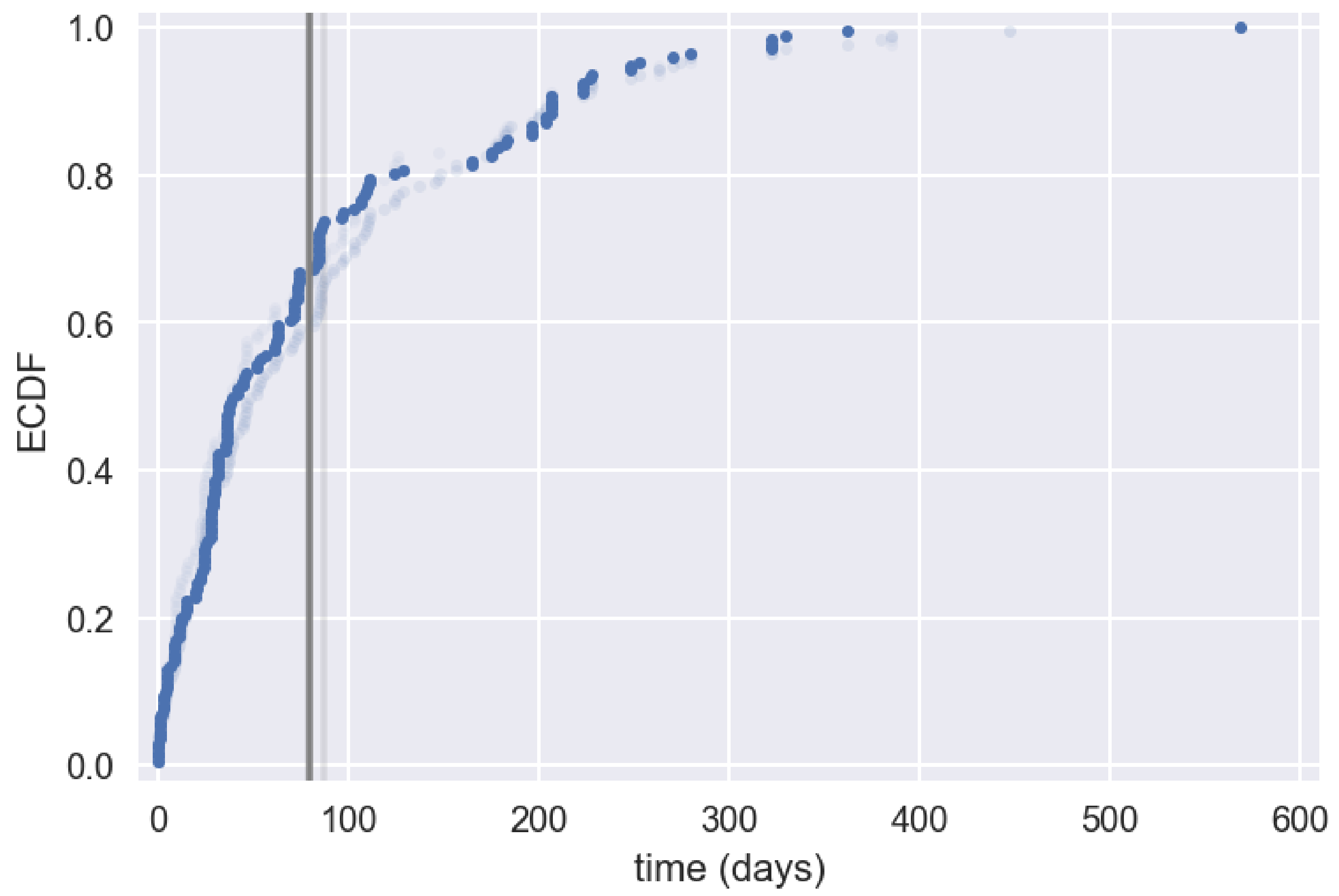
```python
# Resample nuclear_incident_times with replacement
bs_sample = np.random.choice(
  nuclear_incident_times,
  replace=True,
  size=len(inter_times)
)
```

Bootstrap replicates



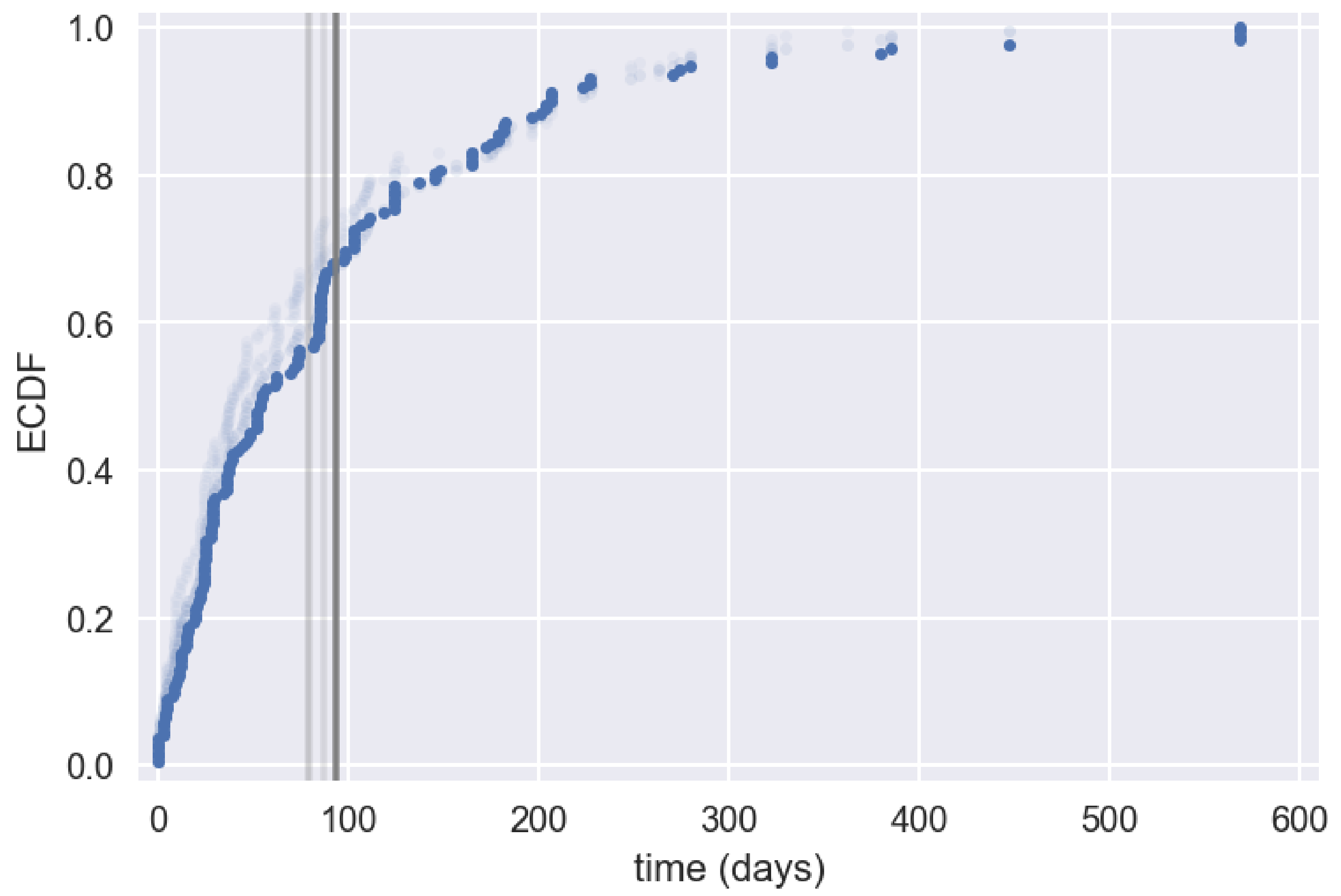
[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

Bootstrap replicates



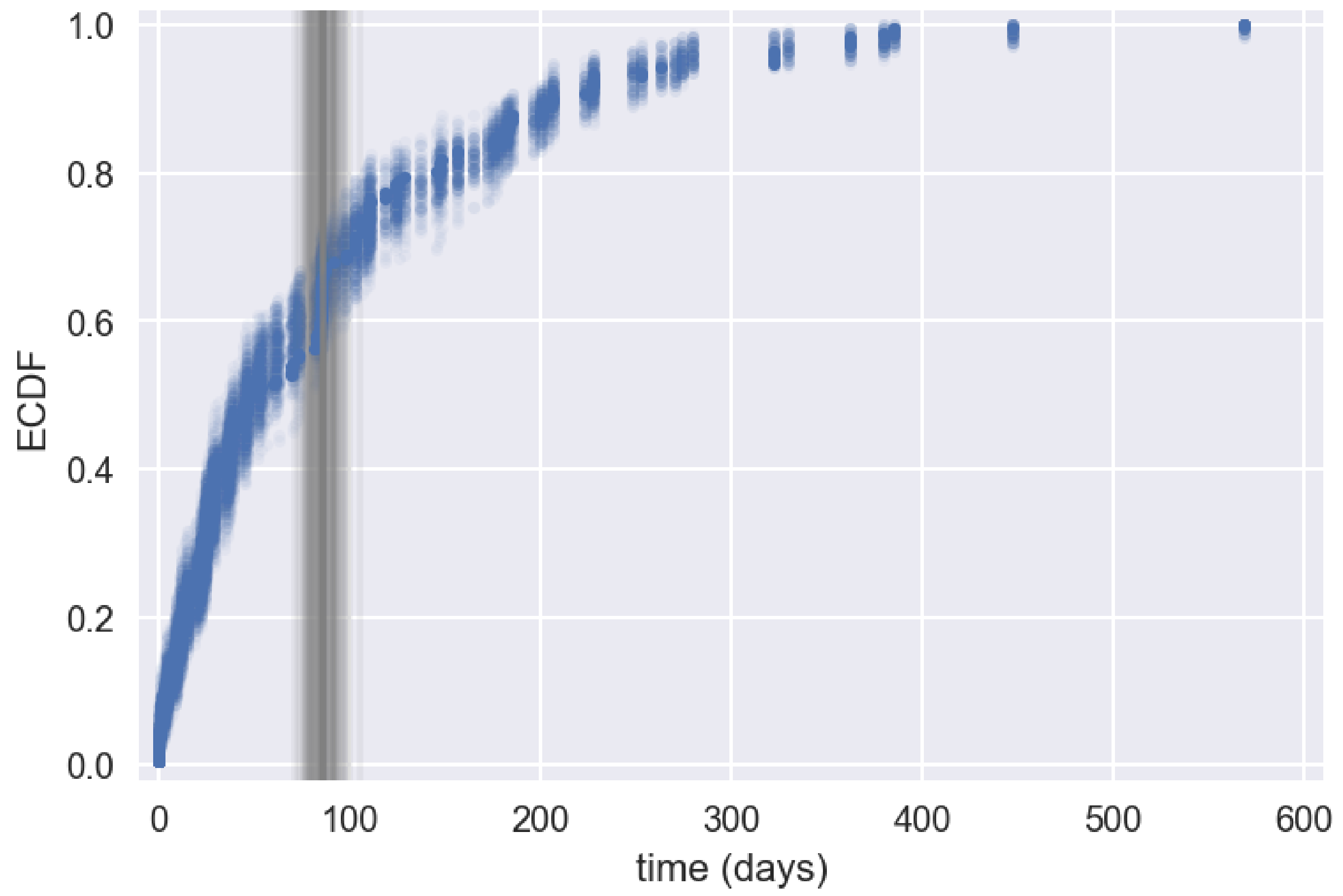
[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

Bootstrap replicates



[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

Bootstrap replicates



[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

Bootstrap replicates

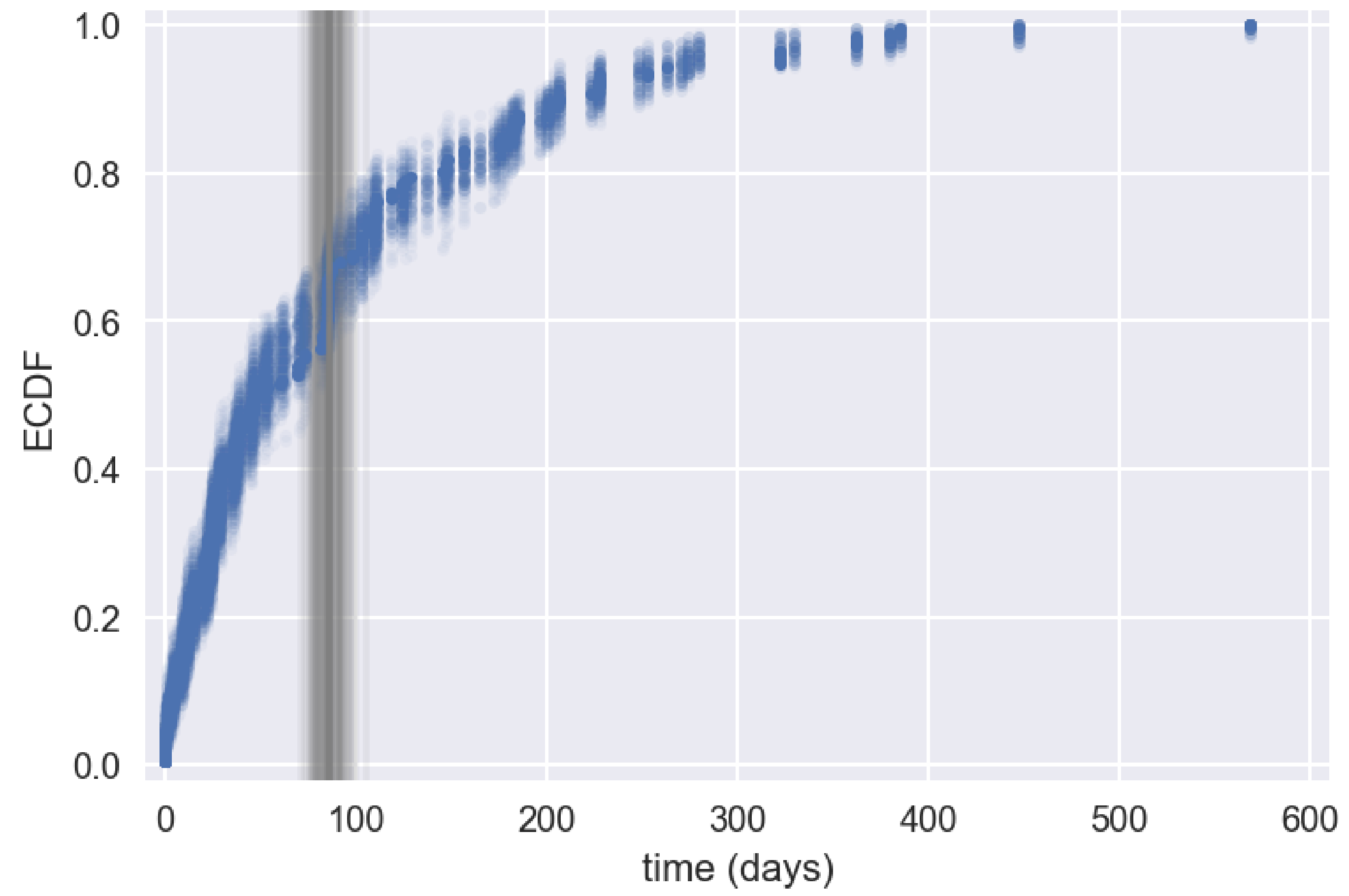
Bootstrap replicate: A statistic computed from a bootstrap sample

dcst.draw_bs_reps()

Function to draw bootstrap replicates from a data set

```python
# Draw 10000 replicates of the mean from
# nuclear_incident_times
bs_reps = dcst.draw_bs_reps(
  nuclear_incident_times, np.mean, size=10000
)
```

The bootstrap confidence interval



[1] Data source: Wheatley, Sovacool, and Sornette, Nuclear Events Database

The bootstrap confidence interval

If we repeated measurements over and over again, *p*% of the observed values would lie within the *p*% confidence interval

The bootstrap confidence interval

```
np.percentile(bs_reps, [2.5, 97.5])
```

```
array([  73.31505848,  102.39181287])
```

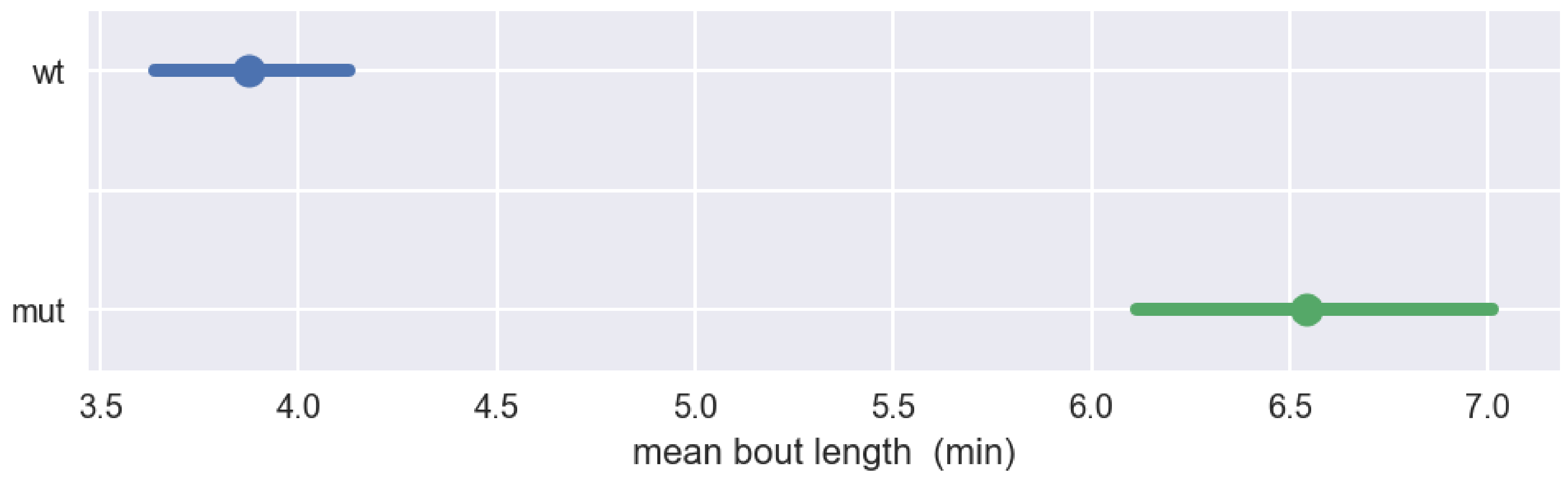
Let's practice!

Hypothesis tests

CASE STUDIES IN STATISTICAL THINKING

Justin Bois
Lecturer, Caltech

Effects of mutation on activity

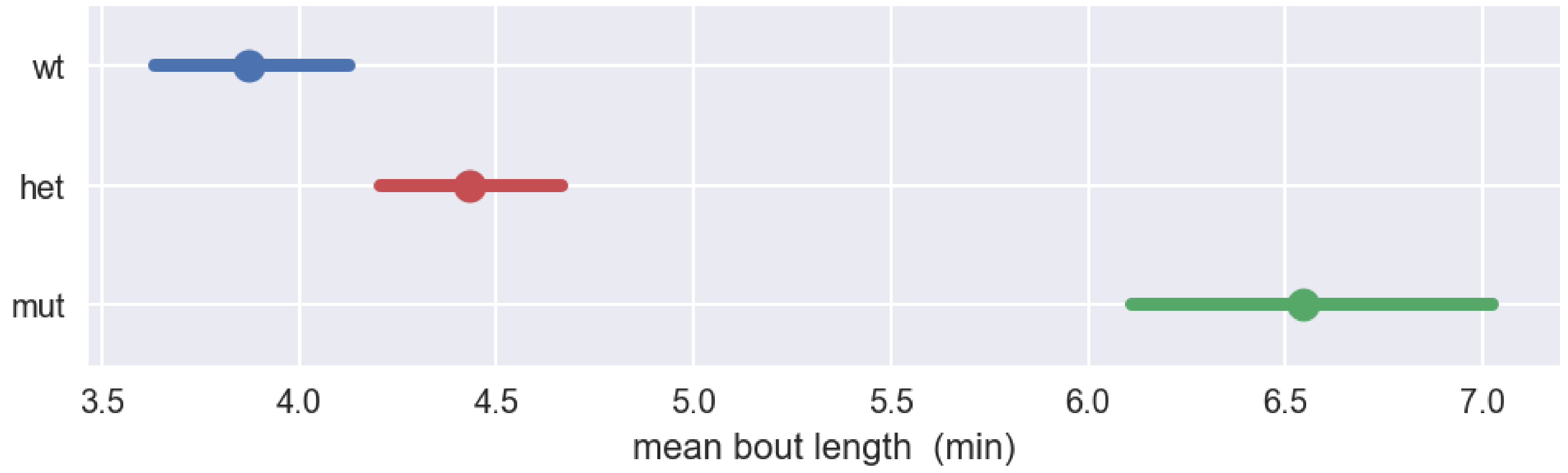


[1] Data courtesy of Avni Gandhi, Grigogios Oikonomou, and David Prober, Caltech

Genotype definitions

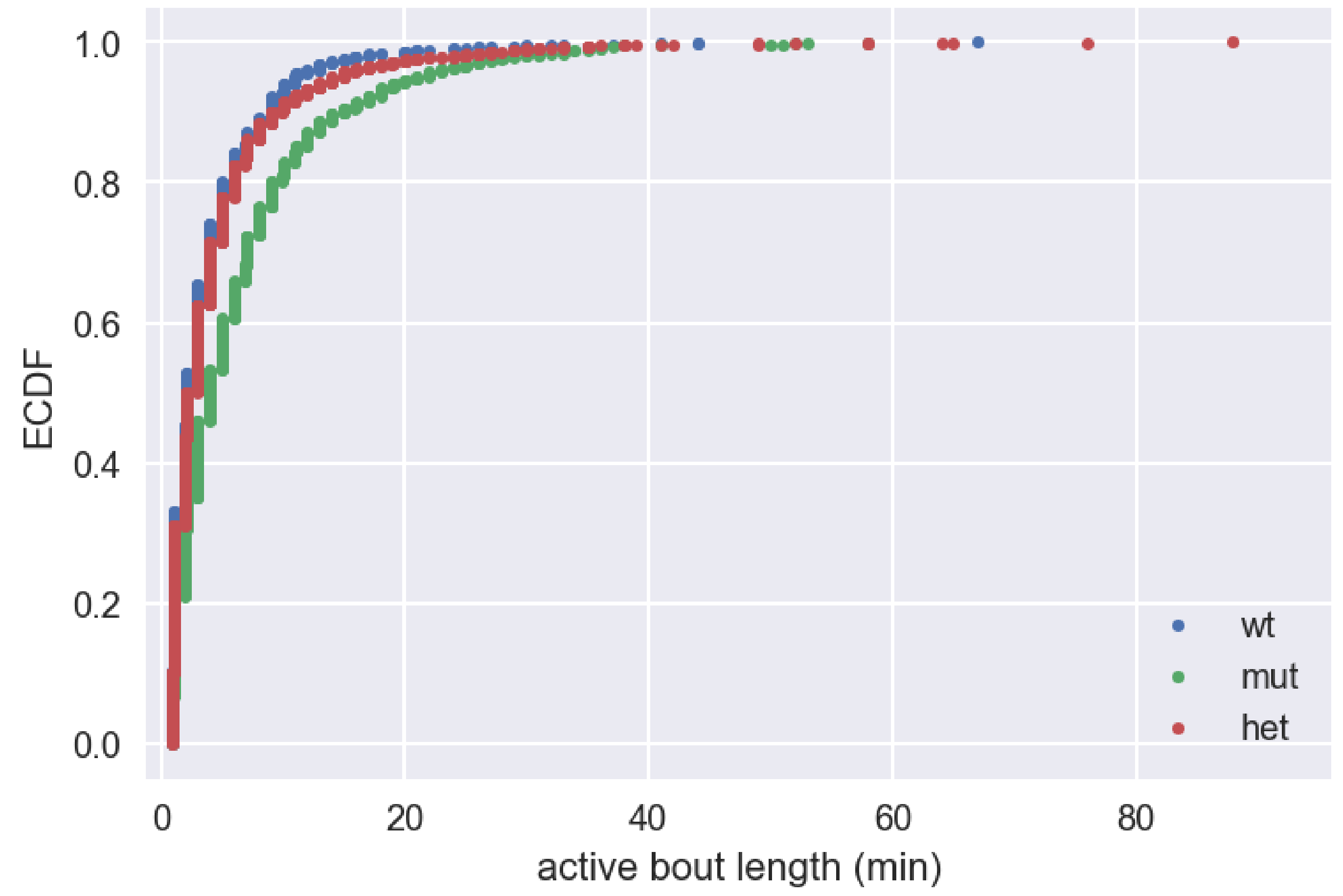
- **Wild type**: No mutations
- **Heterozygote:** Mutation on one of two chromosomes
- **Mutant:** Mutation on both chromosomes

Effects of mutation on activity



[1] Data courtesy of Avni Gandhi, Grigogios Oikonomou, and David Prober, Caltech

Effects of mutation on activity



[1] Data courtesy of Avni Gandhi, Grigogios Oikonomou, and David Prober, Caltech

Hypothesis test

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

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

Test statistic

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

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

Requires clear specification of:

- **Null hypothesis** that can be simulated
- **Test statistic** that can be calculated from observed and simulated data
- Definition of **at least as extreme as**

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

Specifying the test

Null hypothesis: the active bout lengths of wild type and heterozygotic fish are identically distributed

Test statistic: Difference in mean active bout length between heterozygotes and wild type

At least as extreme as: Test statistic is greater than or equal to what was observed

Permutation test

For each replicate:

- Scramble labels of data points
- Compute test statistic

```
perm_reps = dcst.draw_perm_reps(
    data_a, data_b, dcst.diff_of_means, size=10000
)
```

p-value is the fraction of replicates at least as extreme as what was observed

```
p_val = np.sum(perm_reps >= diff_means_obs) / len(perm_reps)
```

Let's practice!

CASE STUDIES IN STATISTICAL THINKING

Linear regressions and pairs bootstrap

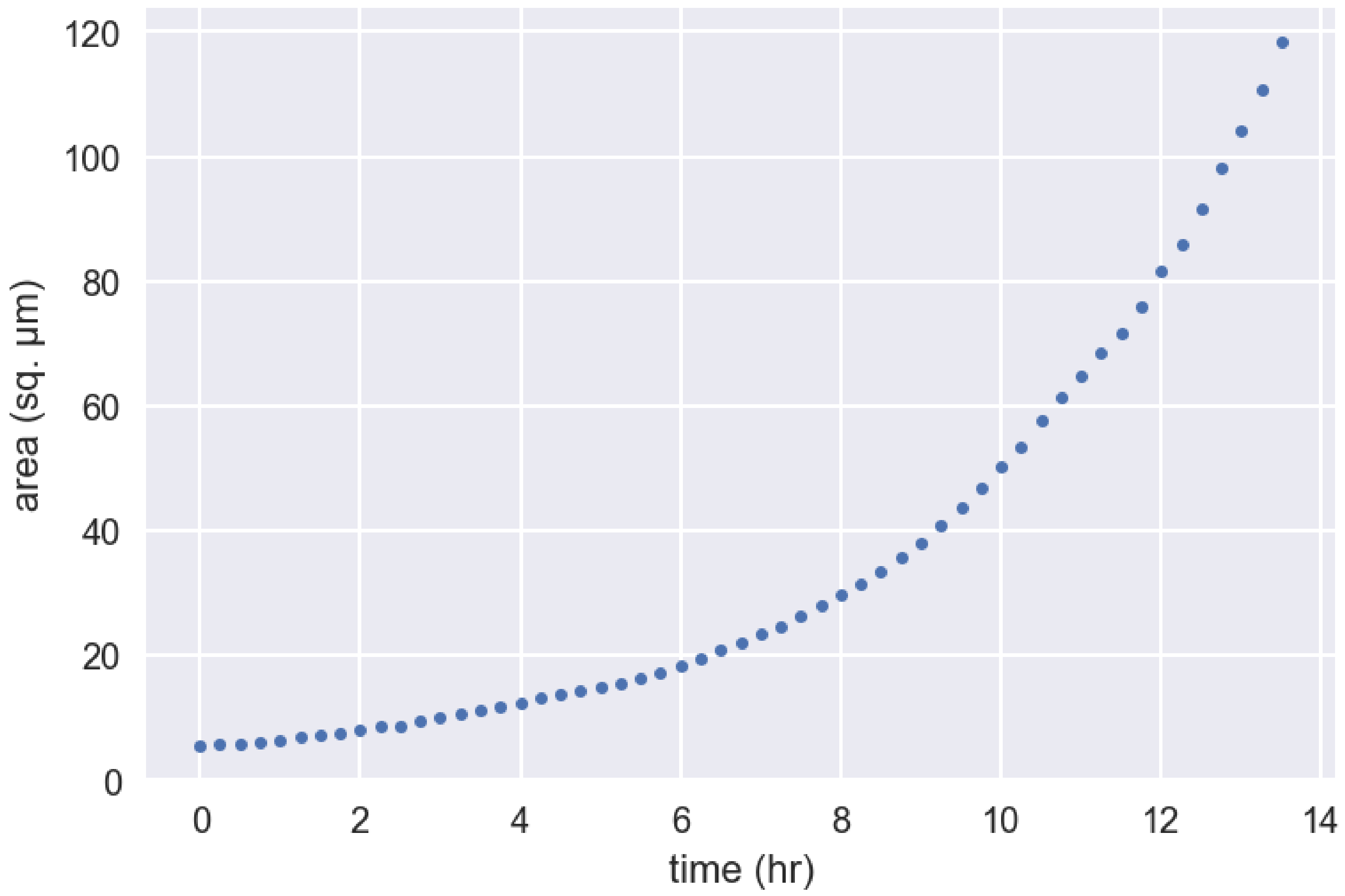
CASE STUDIES IN STATISTICAL THINKING

Justin Bois
Lecturer, Caltech

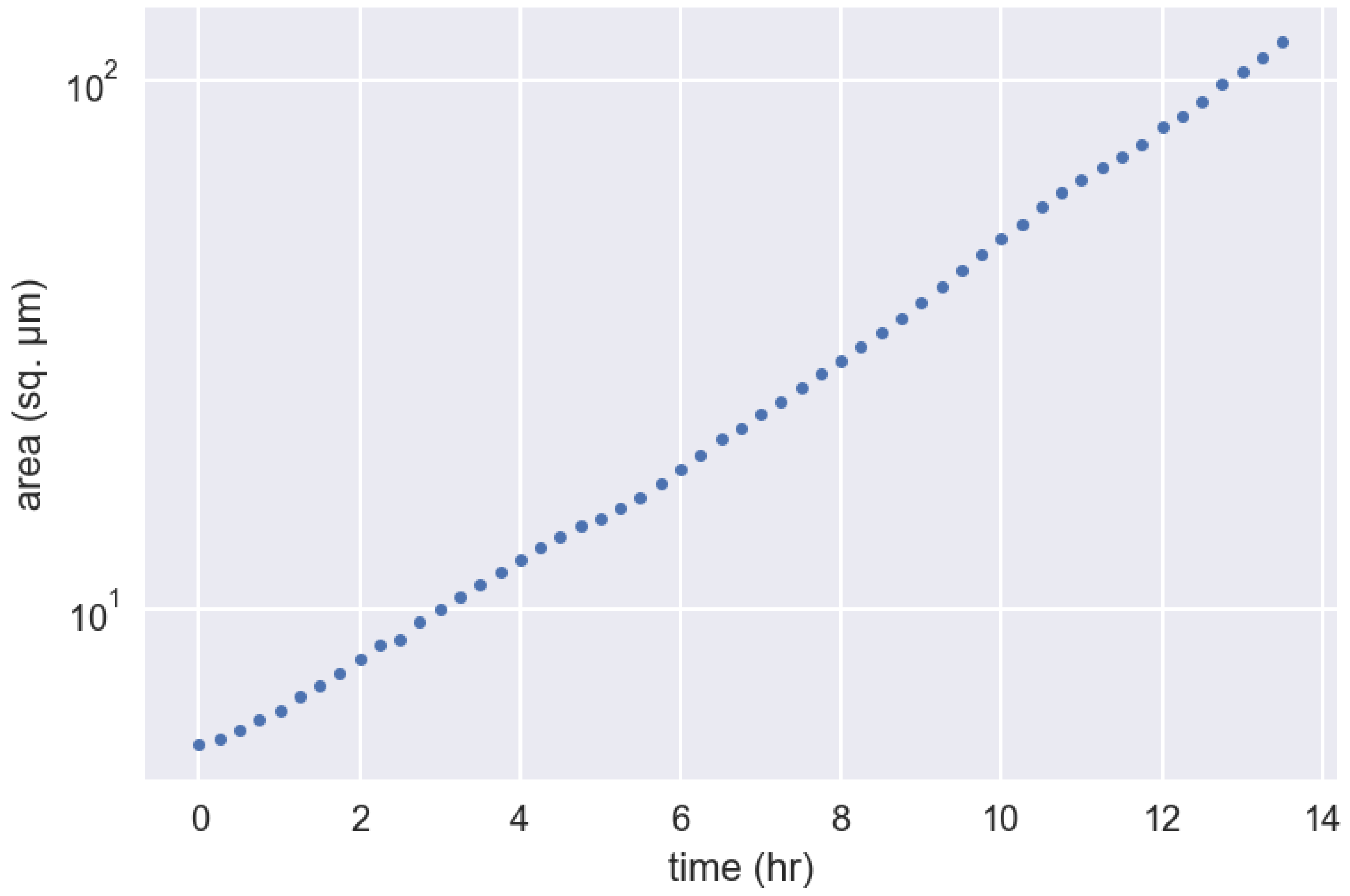
Bacterial growth

[1] Images courtesy of Jin Park and Michael Elowitz, Caltech

Bacterial growth



```
_ = plt.semilogy(t, bac_area, marker='.', linestyle='none')
_ = plt.xlabel('time (hr)')
_ = plt.ylabel('area (sq. µm)')
plt.show()
```



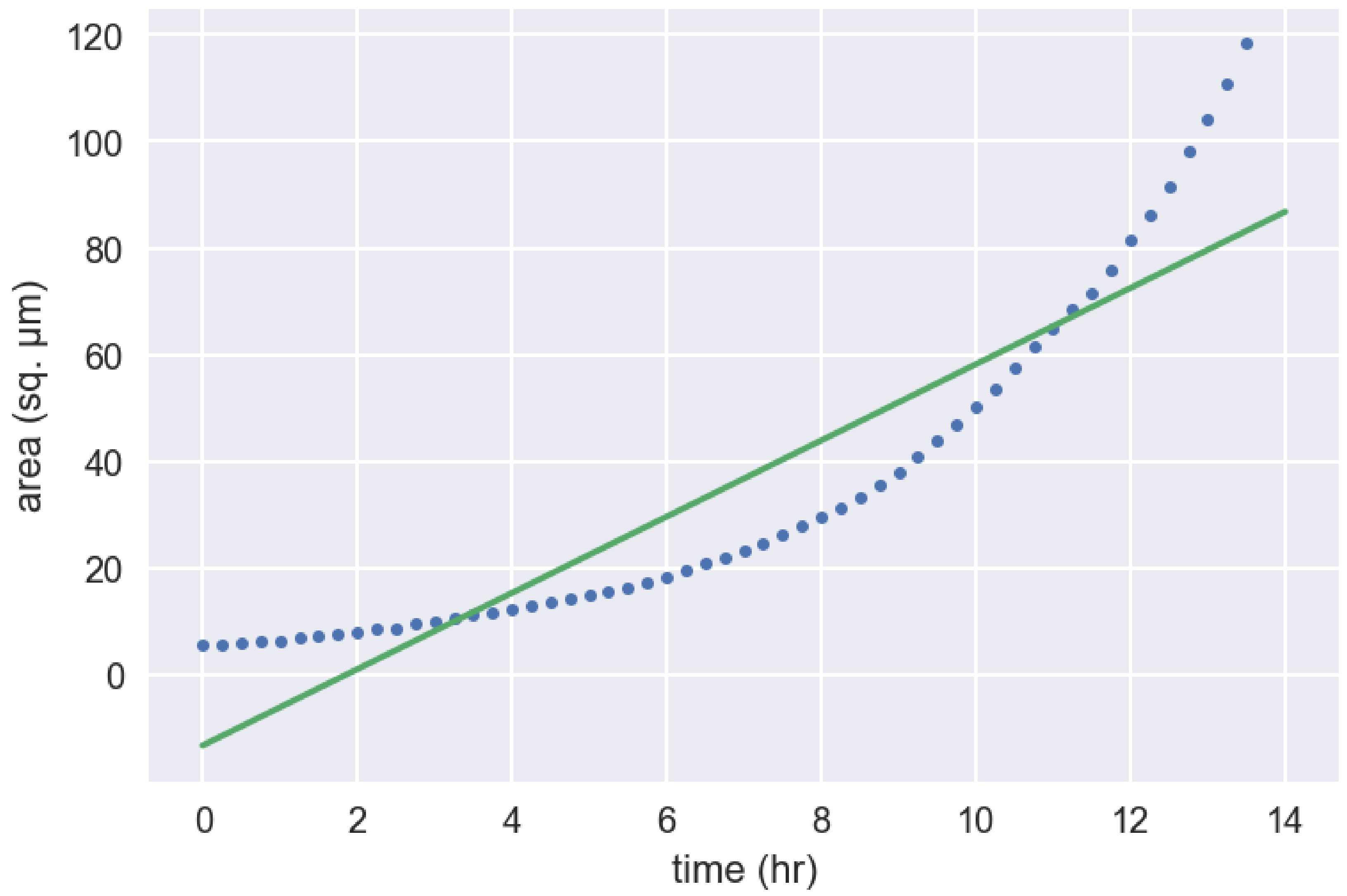
Linear regression with np.polyfit()

```
slope, intercept = np.polyfit(t, bac_area, 1)
```

```
t_theor = np.array([0, 14])
bac_area_theor = slope * t_theor + intercept
```

```
_ = plt.plot(t, bac_area, marker='.', linestyle='none')
_ = plt.plot(t_theor, bac_area_theor)
_ = plt.xlabel('time (hr)')
_ = plt.ylabel('area (sq. µm)')
plt.show()
```

Regression of bacterial growth



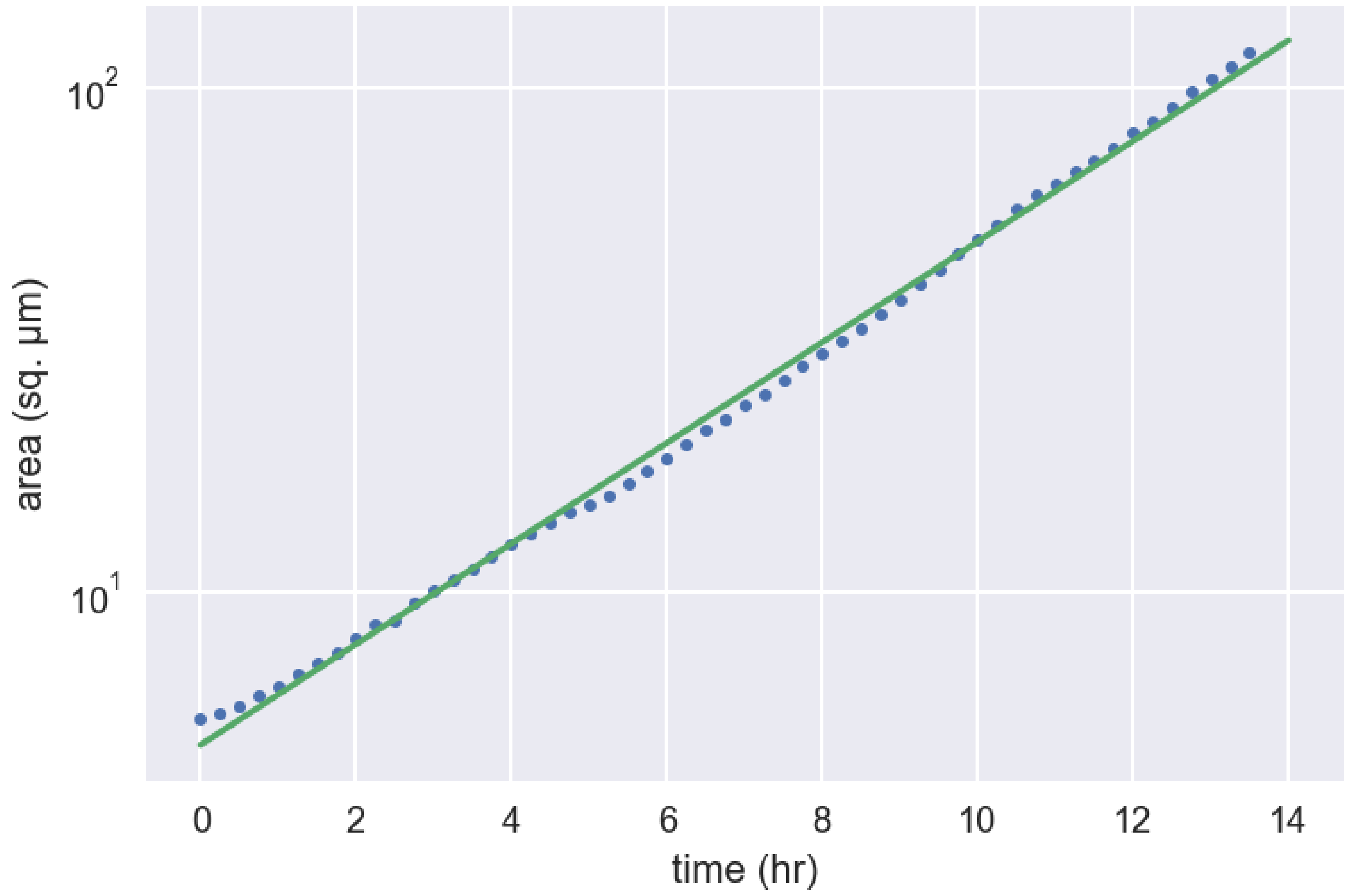
Semilog-linear regression with np.polyfit()

```
slope, intercept = np.polyfit(t, np.log(bac_area), 1)
```

```
t_theor = np.array([0, 14])
bac_area_theor = np.exp(slope * t_theor + intercept)
```

```
_ = plt.semilogy(t, bac_area, marker='.', linestyle='none')
_ = plt.semilogy(t_theor, bac_area_theor)
_ = plt.xlabel('time (hr)')
_ = plt.ylabel('area (sq. µm)')
plt.show()
```

Regression of bacterial growth



Pairs bootstrap

- Resample data in pairs
- Compute slope and intercept from resampled data
- Each slope and intercept is a bootstrap replicate
- Compute confidence intervals from percentiles of bootstrap replicates

Pairs bootstrap

```
# Draw 10000 pairs bootstrap reps
slope_reps, int_reps = dcst.draw_bs_pairs_linreg(
  x_data, y_data, size=10000
)

# Compute 95% confidence interval of slope
slope_conf_int = np.percentile(slope_reps, [2.5, 97.5])
```

Let's practice!

CASE STUDIES IN STATISTICAL THINKING