Probability & Statistics

Lecture 4

Y-DATA course

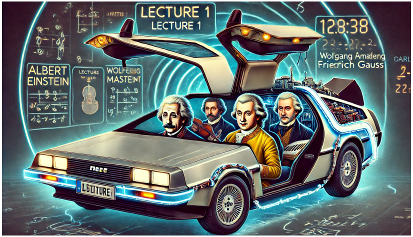
Prof. Daniel Nevo, Tel Aviv University

Fall 2024–2025

Part 1 summary

- **Probability:** Definition, Events, Sets, Probability axioms and rules
- **Discrete random variables:** Definition, PMF, Expectation, Variance.
- **Family of distributions:** Bernoulli, Discrete Uniform, Binomial, Poisson, Geometric
- **Continuous random variables:** Definition, PDF, CDF, Quantiles ,Expectation, Variance, Quantiles,
- **Family of distributions:** Continuous Uniform, Exponential, Normal, Standard Normal
- **Descriptive statistics**

What are we trying to do?



Statistics

- The "meta" science of data: collecting data, summarizing data, interpreting data, and modeling data.
- **Descriptive statistics** describe and summarize data.
- **Statistical modeling** often use probability to approximate real-life processes.
- Use the data to estimate model parameters - the building blocks of the probability model.
- Quantify uncertainty of our estimation and test assumptions about reality using the data.
- The general goal is to learn from **sample** about about the **population**.

Overarching goal

- We want to learn something about the *population*.
    - This could be a well-defined population – all people who currently use our product at least once a week.
    - Or more vague/theoretical – all people who will visit my website in the next year. (Sometimes called super-population)
- If the population is large enough, we often view it as *infinite*.

The General Concept of a Statistical Model

- A **statistical model** describes the relationship between the population and the data we observe.
- The population is associated with a probability distribution. This distribution represents the **true** underlying behavior or characteristics of the population.
- A sample of size n consists of data drawn from this population.
- We assume the sample data are generated according to the same probability distribution as the population.
- The goal is to use the sample data to learn about the population distribution.

The General Concept of a Statistical Model

- A **statistical model** describes the relationship between the population and the data we observe.
- The population is associated with a probability distribution. This distribution represents the **true** underlying behavior or characteristics of the population.
- A sample of size n consists of data drawn from this population.
- We assume the sample data are generated according to the same probability distribution as the population.
- The goal is to use the sample data to learn about the population distribution.
- **A cautionary note:** the way we sample matters.

Parameters

- We think about X as the random variable in the population. We assume something/nothing on its distribution.
- Then we want to learn something about the distribution, often times a *parameter* θ.

Parameters

- We think about X as the random variable in the population. We assume something/nothing on its distribution.
- Then we want to learn something about the distribution, often times a *parameter* θ.
- Examples for such θ,
  - The rate of Exponential distribution. $X \sim Exp(\theta)$
  - The probability for Bernoulli variable. $X \sim Ber(\theta)$.
  - The expectations of a random variable with no assumptions $E[X] = \theta$.

Parameters

- We think about X as the random variable in the population. We assume something/nothing on its distribution.
- Then we want to learn something about the distribution, often times a *parameter* θ.
- Examples for such θ,
  - The rate of Exponential distribution. $X \sim Exp(\theta)$
  - The probability for Bernoulli variable. $X \sim Ber(\theta)$.
  - The expectations of a random variable with no assumptions $E[X] = \theta$.
- Our goal is to **estimate** θ.
- (θ is just a name, we can use whatever Greek letter to denote what we want to estimate).

Tips Dataset

- In one restaurant, a food server recorded data about 244 customers they served during an interval of two and a half months in the early 1990s.

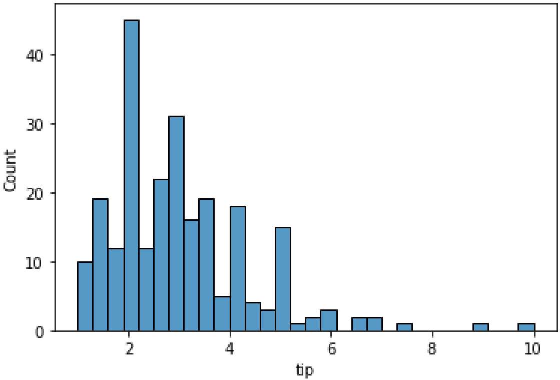
```
import seaborn as sns
import pandas as pd

# Load the dataset
tips = sns.load_dataset("tips")

# Inspect the first few rows
print(tips.head())
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Example: Tips Dataset

- The tip amount is modeled as a random variable in the population X. Our goal is to estimate $\theta = E[X]$.
- Can we estimate θ from the data on 244 tips?
- We have observations $X_1, ..., X_{244}$ all with the same distribution (iid).
- **Notation alert** $\theta = E[X]$ means we just give the expected value of X a name. We did not assume anything about the distribution.

Example: Tips Dataset



Estimating the Expectation of a Random Variable

- Suppose we have a random variable X with an unknown distribution.
- We observe a sample $X_1, X_2, \ldots, X_n$ from the population.
- Goal: Estimate the population mean $\theta = E[X]$. Any suggestions?

Estimating the Expectation of a Random Variable

- Suppose we have a random variable X with an unknown distribution.
- We observe a sample $X_1, X_2, \dots, X_n$ from the population.
- Goal: Estimate the population mean $\theta = E[X]$. Any suggestions?
- A natural **estimator** is the sample mean:

$$\hat{\theta} = \overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Estimating the Expectation of a Random Variable

- Suppose we have a random variable X with an unknown distribution.
- We observe a sample $X_1, X_2, \ldots, X_n$ from the population.
- Goal: Estimate the population mean $\theta = E[X]$. Any suggestions?
- A natural **estimator** is the sample mean:

$$\hat{\theta} = \overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- But wait, is it a good estimator? what's a good estimator anyway?

What is an Estimator?

- **Population parameter:** The true value we aim to estimate (e.g., mean μ, variance σ^2).
- **Estimator:** A rule of function applied to the sample to estimate the parameter.
- **Estimate:** The numerical value produced by the estimator when applied to a sample.

Parameter, Estimator & Estimate

- The parameter is $E[X] = \theta$.
- The estimator is to take the average of the observations, the sample mean $\frac{1}{244}\sum_{i=1}^{244} X_i$.
- The estimate is 3.0.

```
# Calculate the mean tip (rounded 2 decimal places)
mean_tip = round(tips["tip"].mean(), 2)
print(mean_tip).
3.0
```

What is a **Good** Estimator?

- What is a good estimator? What do we expect our estimator to achieve?
- Key issue: an estimator is a **random variable**.

What is a **Good** Estimator?

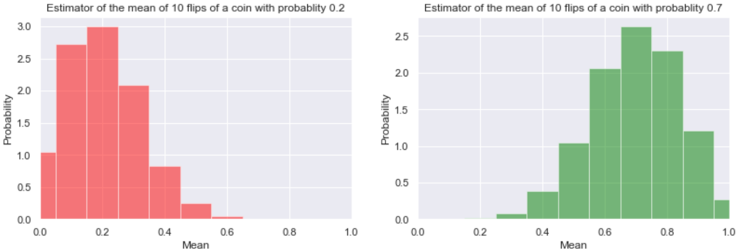
- What is a good estimator? What do we expect our estimator to achieve?
- Key issue: an estimator is a **random variable**.
- Example (543): I have a magic coin with some probability θ for heads.
- I flip the coin 10 times, and want to estimate θ. Makes sense to estimate θ with the empirical proportion of heads.
- Formally, $X_1, ..., X_{10} \sim Ber(\theta)$.
- Our estimator is $\hat{\theta} = \overline{X}_{10}$. What's the support of $\hat{\theta}$?

An Estimator is a Random variable

- What's the distribution of $\hat{\theta}$?

An Estimator is a Random variable

- What's the distribution of $\hat{\theta}$?
- Answer: it depends on the true value of θ.



- This is an example of Monte Carlo simulation.

An Estimator is a Random variable

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
np.random.seed(314)
n = 10
p = 0.7
print(np.random.binomial(n, p, 1)/10)
print(np.random.binomial(n, p, 1)/10)
print(np.random.binomial(n, p, 1)/10)
print(np.random.binomial(n, p, 1)/10)
print(np.random.binomial(n, p, 1)/10)
print(np.random.binomial(n, p, 1)/10)
[0.5]
[0.7]
[0.8]
[0.6]
[0.5]
[0.6]
```

Bias of an Estimator

- A good estimator is an estimator that on average gets the right answer. How to say this formally?
- An estimator $\hat{\theta}$ is **unbiased** if

$$E[\hat{\theta}] = \theta$$

Bias of an Estimator

- A good estimator is an estimator that on average gets the right answer. How to say this formally?
- An estimator $\hat{\theta}$ is **unbiased** if

$$E[\hat{\theta}] = \theta$$

- If $E[\hat{\theta}] \neq \theta$ we say the estimator is **biased**.
- The bias of an estimator is $E[\hat{\theta} - \theta] = E[\hat{\theta}] - \theta$.
  - If $E[\hat{\theta}] - \theta > 0$ the estimator is said to overestimate θ (on average).
  - If $E[\hat{\theta}] - \theta < 0$ the estimator is said to underestimate θ (on average).

The Unbiasedness of the Sample Mean

- A natural **estimator** for the population's expected value is the sample mean:

$$\hat{\theta} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- It is an unbiased estimator! Proof:

$$\begin{aligned} E[\hat{\theta}] &= E\left[\frac{1}{n} \sum_{i=1}^{n} X_i\right] = \frac{1}{n} E\left[\sum_{i=1}^{n} X_i\right] \\ &= \frac{1}{n} \sum_{i=1}^{n} E[X_i] = \frac{1}{n} \sum_{i=1}^{n} \theta \\ &= \frac{1}{n} n\theta = \theta \end{aligned}$$

Bias Versus Error

- Unbiased estimators could still be bad estimators.
- An unbiased estimator just means that on average we're fine. But the error $\hat{\theta} - \theta$ could still be far away from zero.

Bias Versus Error

- Unbiased estimators could still be bad estimators.
- An unbiased estimator just means that on average we're fine. But the error $\hat{\theta} - \theta$ could still be far away from zero.
- For example, the result of the first coin flip is also an unbiased estimator.
- Formally $\hat{\theta} = X_1$. We have $E[\hat{\theta}] = E[X_1] = \theta$, for any θ.
- But X_1 is either one or zero so unless θ is zero or one. This estimator is always wrong.

Variance of an Estimator

- An estimator is a random variable. Bias speaks to its expected value, but what about its spread?
- The variance of an estimator measures the estimator's spread:

$$Var(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2] =$$
$$= E[\hat{\theta}^2] - E[\hat{\theta}^2]$$

- A good estimator will have low bias (on average it's correct) and low variance (it tends to be close to that average).

The Variance of the Sample Mean

- A natural **estimator** for the population's expected value is the sample mean $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} X_i$
- If we have $X_1, ..., X_n$ which are iid, with expectation $E[X] = \theta$ and variance $Var(X) = \sigma^2$.
- The sample mean is an unbiased estimator of θ. Its variance is

$$
\begin{aligned}
Var(\hat{\theta}) = Var\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) &= \frac{1}{n^2} Var\left(\sum_{i=1}^{n} X_i\right) \\
&= \frac{1}{n^2}\sum_{i=1}^{n} Var(X_i) = \frac{1}{n}\sum_{i=1}^{n} \sigma^2 \\
&= \frac{1}{n^2} n\sigma^2 = \frac{\sigma^2}{n}
\end{aligned}
$$

The Variance of the Sample Mean

- Consider $X_1, ..., X_n$ iid random variables, with expectation $E[X] = \theta$ and variance $Var(X) = \sigma^2$.
- Let's compare two estimators: the first result X_1 and the sample mean $\overline{X}_{10} = \frac{1}{10}\sum_{i=1}^{10} X_i$
- The are both **unbiased**. $E[X_1] = E[\overline{X}_{10}] = \theta$

The Variance of the Sample Mean

- Consider $X_1, ..., X_n$ iid random variables, with expectation $E[X] = \theta$ and variance $Var(X) = \sigma^2$.
- Let's compare two estimators: the first result X_1 and the sample mean $\overline{X}_{10} = \frac{1}{10} \sum_{i=1}^{10} X_i$
- The are both **unbiased**. $E[X_1] = E[\overline{X}_{10}] = \theta$
- The variance of X_1 is σ^2.
- The variance of $\overline{X}_{10}$ is $\frac{\sigma^2}{10}$.

The Variance of the Sample Mean

- Consider $X_1, ..., X_n$ iid random variables, with expectation $E[X] = \theta$ and variance $Var(X) = \sigma^2$.
- Let's compare two estimators: the first result X_1 and the sample mean $\overline{X}_{10} = \frac{1}{10} \sum_{i=1}^{10} X_i$
- The are both **unbiased**. $E[X_1] = E[\overline{X}_{10}] = \theta$
- The variance of X_1 is σ^2.
- The variance of $\overline{X}_{10}$ is $\frac{\sigma^2}{10}$.
- No matter what's the true θ, $\overline{X}_{10}$ is a much better estimator.

Mean Squared Error (MSE)

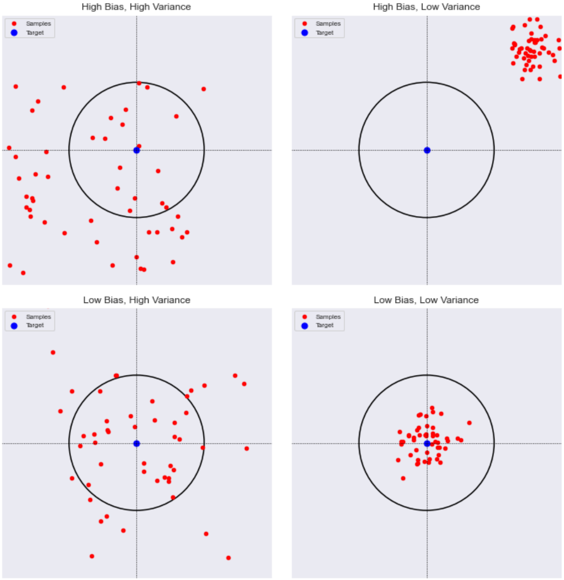
- The mean squared error is $E[(\hat{\theta} - \theta)^2]$.
- When does it equal to the variance? Recall $Var(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$

Mean Squared Error (MSE)

- The mean squared error is $E[(\hat{\theta} - \theta)^2]$.
- When does it equal to the variance? Recall $Var(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$
- Only for unbiased estimators.
- More generally, the MSE combines bias and variance to assess estimator's accuracy.

$$MSE(\hat{\theta}) = Var(\hat{\theta}) + (E[\hat{\theta} - \theta])^2$$

Bias vs. Variance

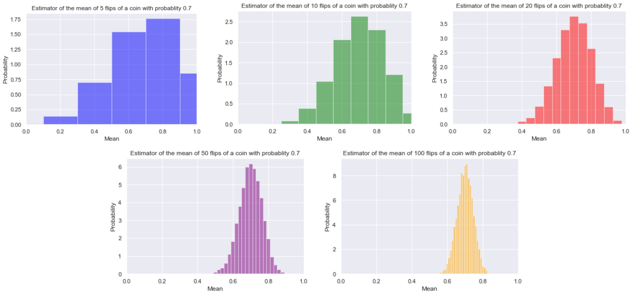


An Estimator is a Random variable

- What if I had flipped the coin 5 times? 20 times?
- The distribution of $\overline{X}_n$ depends on n.

An Estimator is a Random variable

- What if I had flipped the coin 5 times? 20 times?
- The distribution of $\overline{X}_n$ depends on n.



Consistency

- A good estimator is an estimator which gets better and better as the sample size increases.
- An estimator $\hat{\theta}$ is a *consistent estimator* if

$$\Pr(|\hat{\theta}_n - \theta| < \epsilon) \to 1 \quad \text{as } n \to \infty$$

  for any ϵ.
- In words, for any small number ϵ the probability that $\hat{\theta}_n$ is as close to θ as ϵ converges to one.

Consistency

- A good estimator is an estimator which gets better and better as the sample size increases.
- An estimator $\hat{\theta}$ is a *consistent estimator* if

$$\Pr(|\hat{\theta}_n - \theta| < \epsilon) \to 1 \quad \text{as } n \to \infty$$

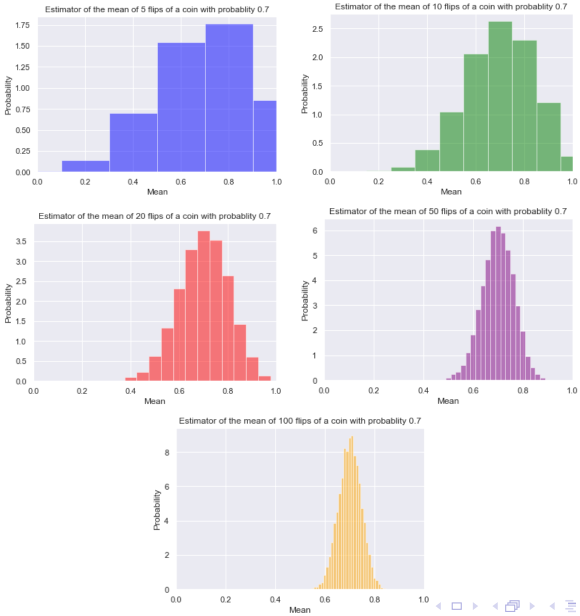
  for any ϵ.
- In words, for any small number ϵ the probability that $\hat{\theta}_n$ is as close to θ as ϵ converges to one.
- Formally this type of convergence is called *convergence in probability*, and we write $\hat{\theta} \xrightarrow{p} \theta$.
- Does the sample mean is a consistent estimator of the expectation?

Law of Large Numbers

- The **Law of Large Numbers (LLN)** states that as the sample size n grows, the sample average $\overline{X}_n$ converges in probability to the true mean θ.
- Let $X_1, ..., X_n$ be an iid sample from a distribution X with expectation $E[X] = \theta$
- Then, for any $\epsilon > 0$:

$$\Pr(|\hat{\theta}_n - \theta| < \epsilon) \to 1 \quad \text{as } n \to \infty.$$

An Estimator is a Random variable



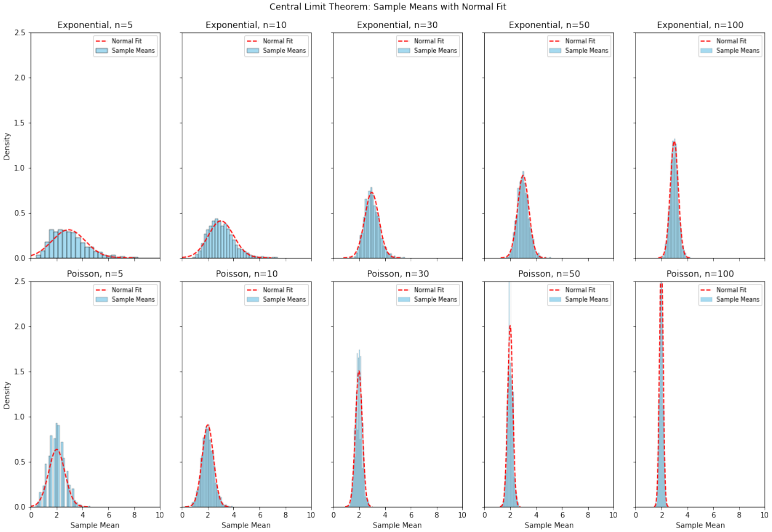
Central Limit Theorem

- Since we are taking the sample mean, it has one more important property.
- The Central Limit Theorem (CLT) states that the sampling distribution of the sample mean $\overline{X}_n$ approaches a normal distribution as n increases. Informally,

$$\overline{X}_n \approx N\left(\mu, \frac{\sigma^2}{n}\right)$$

- **Regardless of the population distribution**, the sample mean's distribution will be approximate normal.
- We already knew about the mean and variance, CLT also tells us the distribution.

CLT in Action



Variance Estimation

- Let $X_1, ..., X_n$ be an iid sample, with $E[X] = \mu$ and $Var(X) = \sigma^2$.
- We have an estimator for μ, $\hat{\mu} = \bar{X}_n$.
- What about estimating σ^2?
- Intuitively, we could have taken $\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X}_n)^2$
- We don't know μ so we replaced it with an unbiased estimator.
- Is this a good estimator?

Variance Estimation

- It turns out

$$E\left[\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2\right] = \frac{n-1}{n}\sigma^2$$

- The sample variance is biased!
- If we divide by $n-1$ instead we get an unbiased estimator.

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2\right] = \sigma^2$$

Variance Estimation

- It turns out

$$E\left[\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2\right] = \frac{n-1}{n}\sigma^2$$

- The sample variance is biased!
- If we divide by $n-1$ instead we get an unbiased estimator.

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2\right] = \sigma^2$$

- So the typical variance estimator is

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$$

Variance Estimation - of The Estimator

- Let $X_1, ..., X_n$ be an iid sample, with $E[X] = \mu$ and $Var(X) = \sigma^2$.
- We have an estimator for μ, $\hat{\mu} = \bar{X}_n$. We know its variance is

$$Var(\bar{X}_n) = \frac{\sigma^2}{n}$$

- But when we see the data, we don't know σ^2.
- We can *estimate* the variance of our estimator. Because $\hat{\sigma}^2 = S^2$, we can provide an estimator

$$\widehat{Var}(\bar{X}_n) = \frac{S^2}{n}$$

so we can assess the uncertainty around our estimator.

Standard Error

- More commonly we report the standard error (SE) of an estimator and not its variance.
- The SE is
$$SE(\hat{\theta}) = \sqrt{Var(\hat{\theta})}$$
which is often estimated by
$$\widehat{SE}(\hat{\theta}) = \sqrt{\widehat{Var}(\hat{\theta})}$$
- The SE is simply the SD **of the estimator**. In the case of $\hat{\theta} = \bar{X}_n$, the SD is σ, the SE is $\sigma/\sqrt{n}$.
- More generally, $SE(\hat{\theta})$ is **not** always $SD/\sqrt{n}$.

Tips Example

```
mean_tip = tips["tip"].mean()
# Estimate the SE of our estimator of the mean tip
n = tips.shape[0]
var_mean_tip = tips["tip"].var() / n
SE_mean_tip = var_mean_tip ** 0.5

# Print mean tip and standard error
print('Estimate of the expected tip amount: ', round(mean_tip, 2))
print('Standard error of the estimate: ', round(SE_mean_tip, 3))
```

```
Estimate of the expected tip amount:  3.0
Standard error of the estimate:  0.089
```

Estimating a Proportion

- **Goal:** Estimate the true proportion p of a population based on a sample.
- $X_1, ..., X_n$ are iid $Ber(p)$.
- **Point Estimate:** The sample proportion.

$$\widehat{p} = \frac{\sum_{i=1}^{n} X_i}{n}$$

- **SE estimator:** Because proportion is just a mean, and because $Var(X_i) = p(1-p)$ for Bernoulli variables then

$$\widehat{SE}(\hat{p}) = \frac{\widehat{p}(1-\widehat{p})}{n}$$

Tips Example

- Here we compare two estimators for $\Pr(X > 5)$ where X is the tip. First only assuming tips are iid, then adding an assumption of normality

```
# Estimate the probablity of tipping more than 5
p = (tips["tip"] > 2).mean()
print(’Probability of tipping more than $5: ’, round(p, 3))

# Now estimate the same probablity using a normal model
from scipy.stats import norm
mean_tip = tips["tip"].mean()
std_tip = tips["tip"].std()
print(’Mean tip: ’, round(mean_tip, 2))
print(’Standard deviation of tip: ’, round(std_tip, 2))
p_normal = 1 - norm.cdf(5, mean_tip, std_tip)
print(’Probability of tipping more than $5 using normal model: ’,
round(p_normal, 3))
```

```
Probability of tipping more than $5:  0.074
Mean tip:  3.0
Standard deviation of tip:  1.38
Probability of tipping more than $5 using normal model:  0.074
```

Tips Example

- Here we compare two estimators for $\Pr(X < 2)$ where X is the tip. First only assuming tips are iid, then adding an assumption of normality

```
# Repeat the same for the probability of tipping less than 2
p = (tips["tip"] < 2).mean()
print('Probability of tipping less than $2: ', round(p, 3))
p_normal = norm.cdf(2, mean_tip, std_tip)
print('Probability of tipping less than $2 using normal model: ',
round(p_normal, 3))
```

```
Probability of tipping less than $2:  0.184
Probability of tipping less than $2 using normal model:  0.235
```

Finding Estimators

- So far we discussed desirable properties of potential estimators, and considered mainly the sample mean as a natural estimator of the expectation.
- Now we will discuss two general methods to find estimators, given more assumptions.
- For example, that we are willing to assume the family of distributions bit we don't known the parameter.
- The first one is called the *method of moments*.

Method of Moments

- The method of moments is a technique for parameter estimation based on matching sample moments to population moments.
- The k-th moment of a random variable X is defined as $E[X^k]$.
- For example, the first moment is the mean, $E[X]$, and the second central moment is the variance $E[X^2]$.

Method of Moments

- The method of moments is a technique for parameter estimation based on matching sample moments to population moments.
- The k-th moment of a random variable X is defined as $E[X^k]$.
- For example, the first moment is the mean, $E[X]$, and the second central moment is the variance $E[X^2]$.
- The sample k-moment is $\overline{X}_n^k = \frac{1}{n}\sum_{i=1}^n X_i^k$.

Method of Moments

- The method of moments is a technique for parameter estimation based on matching sample moments to population moments.
- The k-th moment of a random variable X is defined as $E[X^k]$.
- For example, the first moment is the mean, $E[X]$, and the second central moment is the variance $E[X^2]$.
- The sample k-moment is $\overline{X}_n^k = \frac{1}{n} \sum_{i=1}^{n} X_i^k$.
- **Rationale:** $\overline{X}_n^k$ is a good estimator of $E[X^k]$.
- Set these equal to the theoretical moments expressed in terms of the parameters to estimate.
- Solve the resulting equations for the parameters.

Method of Moments: Example (1)

- $X_1, ..., X_n \sim \text{Binomial}(m, p)$ (iid). The number of trials m is known, p is not.
- Mean: $E[X] = mp$
- Calculating the Method of Moments estimator

$$E[X] = mp$$

$$\overline{X}_n = mp$$

$$\widehat{p} = \frac{\bar{X}_n}{m}$$

Method of Moments: Example (2)

- $X_1, ..., X_n \sim \mathcal{N}(\mu, \sigma^2)$ (iid). Both parameters are unknown.
  - First moment (mean): $E[X] = \mu$.
  - Second moment: $E[X^2] = \mu^2 + \sigma^2$.

Method of Moments: Example (2)

- $X_1, \dots, X_n \sim \mathcal{N}(\mu, \sigma^2)$ (iid). Both parameters are unknown.
  - First moment (mean): $E[X] = \mu$.
  - Second moment: $E[X^2] = \mu^2 + \sigma^2$.

- Method of Moments Estimation:
  - From the first moment:

$$\overline{X}_n = \mu \implies \hat{\mu} = \overline{X}_n$$

  - From the second moment:

$$\overline{X^2}_n = \frac{1}{n}\sum_{i=1}^{n} X_i^2 = \mu^2 + \sigma^2 \implies \hat{\sigma}^2 = \overline{X^2}_n - \bar{X}_n^2$$

which equals ot the sample variance (with n, not $n-1$).

Maximum Likelihood Estimation

- The most common method is maximum likelihood estimation (MLE).
- Say we flipped my magic coin 10 times, in 7 cases we got heads.
- $X_1, ..., X_10 \sim Ber(p)$. We want to estimate p.
- Person 1: my guess is $\hat{p}_1 = 0.2$
- Person 2: my guess is $\hat{p}_2 = 0.5$
- Which guess looks like a better guess to you? and why?

MLE - Rationale

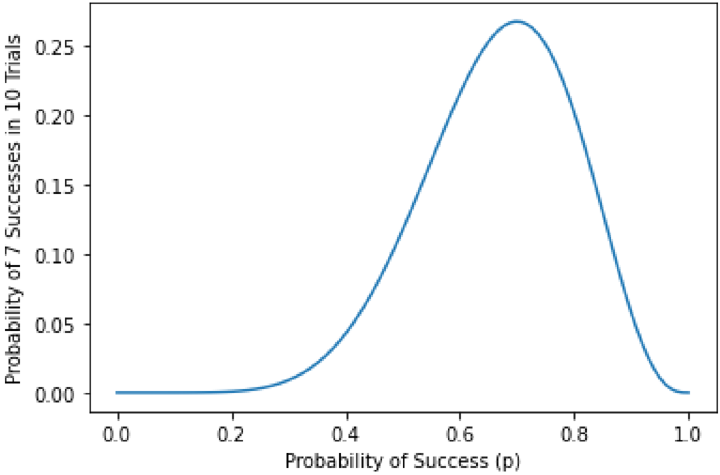
```
# Calculate the probablity of 7 successes in 10 trials,
# with a probability of success of 0.2 and of 0.5

from scipy.stats import binom
import numpy as np

prob1 = binom.pmf(7, 10, 0.2)
print(prob1)
prob2 = binom.pmf(7, 10, 0.5)
print(prob2)
0.000786432
0.11718749999999999
```

- Person 2's guess makes more sense. If $p = 0.2$ the probability of seeing 7 successes is so low that it does not make sense that $p = 0.2$.

MLE - Rationale

```
# Calculate the probablity of 7 successes in 10 trials,
# with a probability of success of 0.2 and of 0.5

from scipy.stats import binom
import numpy as np

prob1 = binom.pmf(7, 10, 0.2)
print(prob1)
prob2 = binom.pmf(7, 10, 0.5)
print(prob2)
0.000786432
0.11718749999999999
```

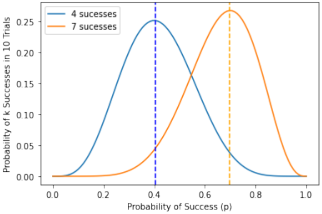
- Person 2's guess makes more sense. If $p = 0.2$ the probability of seeing 7 successes is so low that it does not make sense that $p = 0.2$.
- Person 3: my guess is $\hat{p}_3 = 0.7$
- Person 4: my guess is $\hat{p}_4 = 0.9$
- What do you think?



- What's your guess for p?

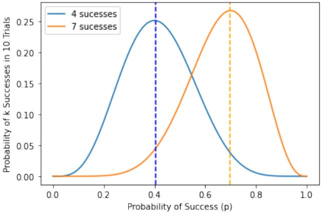
MLE

- This function is called the **likelihood function**,
- The **Maximum Likelihood Estimator** takes the value of the parameter that maximizes this function.
- Essentially, this is the value that makes the probability to observe our sample to be the largest.



MLE

- This function is called the **likelihood function**,
- The **Maximum Likelihood Estimator** takes the value of the parameter that maximizes this function.
- Essentially, this is the value that makes the probability to observe our sample to be the largest.
- Our derivation for the specific sample was ad-hoc, we sometimes can find closed-form formula.



MLE

- The MLE maximizes the likelihood of observing the data given the parameter.

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} L(\theta; X_1, X_2, \ldots, X_n)$$

- $L(\theta) = L(\theta; X_1, X_2, \ldots, X_n)$ is called the likelihood function - it's a function of the parameter given the data.

Example: MLE for Bernoulli Probability

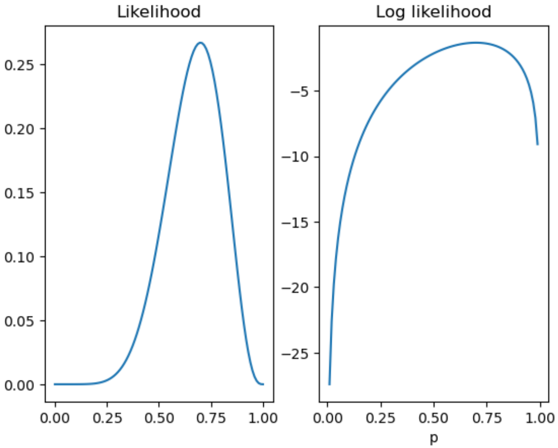
- Let $X_1, X_2, \ldots, X_n$ be i.i.d. Bernoulli random variables with success probability p.
- PMF:
  $\Pr(X_i = x_i) = p^{x_i}(1-p)^{1-x_i},\ x_i \in \{0,1\},\ 0 < p < 1.$
- **Step 1:** Write the likelihood function

$$L(p) = \prod_{i=1}^{n} p^{X_i}(1-p)^{1-X_i} = p^{\sum_{i=1}^{n} X_i}(1-p)^{n-\sum_{i=1}^{n} X_i}$$

MLE for Bernoulli Probability: Log-Likelihood

- **Step 2:** Take log. This is a very common step. Whichever value maximizing L, also maximize $logL$

$$\ell(p) = \log L(p) = \sum_{i=1}^{n} X_i \log p + (n - \sum_{i=1}^{n} X_i) \log(1-p).$$



MLE for Bernoulli Variables: Log-Likelihood

$$\ell(p) = \log L(p) = \sum_{i=1}^{n} X_i \log p + (n - \sum_{i=1}^{n} X_i) \log(1 - p).$$

- **Step 3:** To find a maximum, take the derivative with respect to p and compare to zero.

$$\frac{d\ell(p)}{dp} = \frac{\sum_{i=1}^{n} X_i}{p} - \frac{n - \sum_{i=1}^{n} X_i}{1 - p} = 0$$

$$\Longrightarrow \sum_{i=1}^{n} X_i(1 - p) = (n - \sum_{i=1}^{n} X_i)p$$

$$\Longrightarrow \hat{p} = \frac{\sum_{i=1}^{n} X_i}{n} = \overline{X}_n$$

MLE for Bernoulli Variables: Log-Likelihood

$$\ell(p) = \log L(p) = \sum_{i=1}^{n} X_i \log p + (n - \sum_{i=1}^{n} X_i) \log(1-p).$$

- **Step 3:** To find a maximum, take the derivative with respect to p and compare to zero.

$$\frac{d\ell(p)}{dp} = \frac{\sum_{i=1}^{n} X_i}{p} - \frac{n - \sum_{i=1}^{n} X_i}{1-p} = 0$$

$$\Longrightarrow \sum_{i=1}^{n} X_i(1-p) = (n - \sum_{i=1}^{n} X_i)p$$

$$\Longrightarrow \hat{p} = \frac{\sum_{i=1}^{n} X_i}{n} = \overline{X}_n$$

- (Formally, we need to look a the second derivative to make sure we found the maximum. There are also other technical concerns).

MLE and Method of Moments

- For continuous random variables, we take the density function instead of the PMF (not illustrated here).
- Both the method of moments and the MLE share desirable properties.

MLE and Method of Moments

- For continuous random variables, we take the density function instead of the PMF (not illustrated here).
- Both the method of moments and the MLE share desirable properties.
- **Consistency:** They are both (generally) consistent (but not necessarily unbiased!).
- **Asymptotic distribution:** They both have asymptotic normal distribution (we can even estimate the variance of our estimator).
- **Efficiency:** The MLE has the best (lowest) possible asymptotic variance.

Why Do We Need Confidence Intervals?

- Point estimates $\hat{\theta}$ provide a single value but no information about variability or uncertainty.
- In reality we often don't need to confine ourselves to a single "guess".
- Estimating $SE(\hat{\theta})$ is often possible and desirable, but is not always informative enough.

Why Do We Need Confidence Intervals?

- Point estimates $\hat{\theta}$ provide a single value but no information about variability or uncertainty.
- In reality we often don't need to confine ourselves to a single "guess".
- Estimating $SE(\hat{\theta})$ is often possible and desirable, but is not always informative enough.
- This motivates the idea of **Confidence Intervals**. They
  - Account for sampling variability.
  - Help quantify uncertainty in our estimates.
  - Provide a range of likely values for the true parameter.

Confidence Intervals: Definition

- A **confidence interval** $[C_L, C_U]$ at a *confidence level* $1 - \alpha$ is an interval created such that the proportion of intervals that would contain the true parameter in repeated sampling is $1 - \alpha$
- A 95% confidence interval means we are 95% confident the interval contains the true parameter.

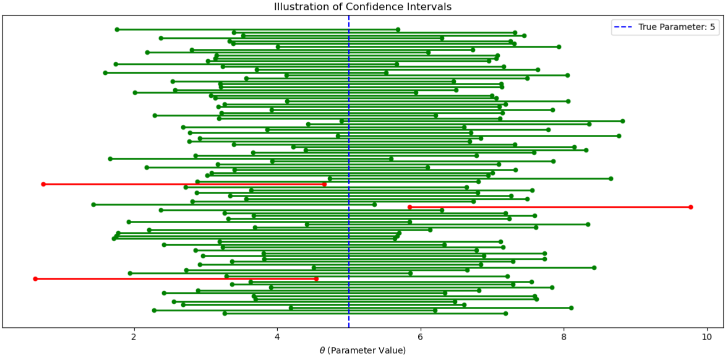
Confidence Intervals: Definition

- A **confidence interval** $[C_L, C_U]$ at a *confidence level* $1 - \alpha$ is an interval created such that the proportion of intervals that would contain the true parameter in repeated sampling is $1 - \alpha$
- A 95% confidence interval means we are 95% confident the interval contains the true parameter.
- It does **not** mean there is a 95% chance the parameter is in the interval.
- The parameter is **fixed**, what's random is the confidence interval itself.

Confidence Intervals: Definition

- A **confidence interval** $[C_L, C_U]$ at a *confidence level* $1-\alpha$ is an interval created such that the proportion of intervals that would contain the true parameter in repeated sampling is $1-\alpha$
- A 95% confidence interval means we are 95% confident the interval contains the true parameter.
- It does **not** mean there is a 95% chance the parameter is in the interval.
- The parameter is **fixed**, what's random is the confidence interval itself.
- The correct statement is that 95% of our confidence intervals constructed with 95% confidence level will contain the true parameter.

Confidence Intervals: Illustration



- Each green line is a confidence interval calculated from a single sample, all with the same parameter.

Constructing CIs

- Let $X_1, ..., X_n$ be iid with $E[X] = \mu$ and $Var(X) = \sigma^2$. Start with the case σ^2 is known.
- Thanks to CLT, if we have that for large enough n, $\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$ (approximately). Therefore,

$$\Pr\left(z_{0.025} \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z_{0.975}\right) = 0.95,$$

where $z_{0.025}$ and $z_{0.975}$ are the 2.5% and 97.5% quantiles of the standard normal distribution.

Constructing CIs

- Let $X_1, ..., X_n$ be iid with $E[X] = \mu$ and $Var(X) = \sigma^2$. Start with the case σ^2 is known.
- Thanks to CLT, if we have that for large enough n, $\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$ (approximately). Therefore,

$$\Pr\left(z_{0.025} \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z_{0.975}\right) = 0.95,$$

  where $z_{0.025}$ and $z_{0.975}$ are the 2.5% and 97.5% quantiles of the standard normal distribution.
- Because of symmetry, $z_{0.025} = -z_{0.975}$ and

$$\Pr\left(-z_{0.975} \leq \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq z_{0.975}\right) = 0.95.$$

Constructing CIs

$$\Pr\left(-z_{0.975} \leq \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \leq z_{0.975}\right) = 0.95.$$

- Rearranging to isolate μ:

$$\Pr\left(\bar{X} - z_{0.975} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{0.975} \cdot \frac{\sigma}{\sqrt{n}}\right) = 0.95$$

- And that's it. We got an interval $[C_L, C_U]$ around μ such that includes μ with probability of 95%

Constructing CIs

- More generally,

$$\Pr\left(\bar{X} - z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}\right) \approx 1 - \alpha$$

  where $z_{1-\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal distribution.

Constructing CIs

- More generally,

$$\Pr\left(\bar{X} - z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}\right) \approx 1 - \alpha$$

  where $z_{1-\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal distribution.

- When the confidence interval is symmetric, we often write it as

$$CI = \bar{X} \pm z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$$

Constructing CIs

$$CI = \bar{X} \pm z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$$

- What affects the width of this confidence interval?

Constructing CIs

$$CI = \bar{X} \pm z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$$

- What affects the width of this confidence interval?
- Larger sample sizes reduce interval width (more precise interval estimator).
- Higher confidence levels increase interval width (more conservative interval estimator).

Tips Example

```
# Calculate 95% confidence interval for the mean tip using
the normal distribution
alpha = 0.05
z = norm.ppf(1 - alpha / 2)
CI95 = (mean_tip - z * SE_mean_tip, mean_tip + z * SE_mean_tip)
print('95% confidence interval for the mean tip: ',
(round(CI95[0], 2), round(CI95[1], 2)))
```

```
95% confidence interval for the mean tip:  (2.82, 3.17)
```

Tips Example

```
# Calculate 90% confidence interval for the mean tip
using the normal distribution
alpha = 0.1
z = norm.ppf(1 - alpha / 2)
CI90 = (mean_tip - z * SE_mean_tip, mean_tip + z * SE_mean_tip)

# Calculate 99% confidence interval for the mean tip
using the normal distribution
alpha = 0.01
z = norm.ppf(1 - alpha / 2)
CI99 = (mean_tip - z * SE_mean_tip, mean_tip + z * SE_mean_tip)

print('90% confidence interval for the mean tip: ',
(round(CI90[0], 2), round(CI90[1], 2)))
print('95% confidence interval for the mean tip: ',
(round(CI95[0], 2), round(CI95[1], 2)))
print('99% confidence interval for the mean tip: ',
(round(CI99[0], 2), round(CI99[1], 2)))
```

```
90% confidence interval for the mean tip:  (2.85, 3.14)
95% confidence interval for the mean tip:  (2.82, 3.17)
99% confidence interval for the mean tip:  (2.77, 3.23)
```

Constructing CIs - Unknown σ^2

- If σ^2 is unknown but n is large we can use the same formula but replace σ with the estimator S.

$$\Pr\left(\bar{X} - z_{1-\alpha/2} \cdot \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1-\alpha/2} \cdot \frac{S}{\sqrt{n}}\right) \approx 1 - \alpha$$

- Intuitively, this is valid because S^2 is a consistent estimator of σ^2.
- If n is not large, but $X_1, \ldots, X_n$ are by themselves approximately $N(\mu, \sigma^2)$ then we still have $\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$ **for any** n

Constructing CIs - Unknown σ^2

- If n is not large, but $X_1, ..., X_n$ are $N(\mu, \sigma^2)$ then we still have $\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$.

  $$\Pr\left(\bar{X} - z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}\right) = 1 - \alpha$$

Constructing CIs - Unknown σ^2

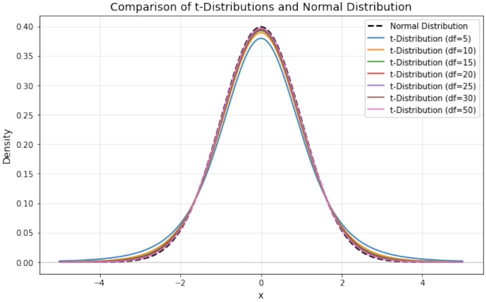
- If n is not large, but $X_1, ..., X_n$ are $N(\mu, \sigma^2)$ then we still have $\bar{X} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$.

$$\Pr\left(\bar{X} - z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}\right) = 1 - \alpha$$

- If σ^2 is unknown we replace σ with S, but it turns out we need to account for this estimation, using the student's t distribution with $n - 1$ degrees of freedom.

$$\Pr\left(\bar{X} - t_{1-\alpha/2,n-1} \cdot \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X} + t_{1-\alpha/2,n-1} \cdot \frac{S}{\sqrt{n}}\right) = 1 - \alpha$$

Comparison Normal Versus t




```
Quantiles for t-distributions:

df=5: 2.5% quantile = -2.5706, 97.5% quantile = 2.5706
df=10: 2.5% quantile = -2.2281, 97.5% quantile = 2.2281
df=15: 2.5% quantile = -2.1314, 97.5% quantile = 2.1314
df=20: 2.5% quantile = -2.0860, 97.5% quantile = 2.0860
df=25: 2.5% quantile = -2.0595, 97.5% quantile = 2.0595
df=30: 2.5% quantile = -2.0423, 97.5% quantile = 2.0423
df=50: 2.5% quantile = -2.0086, 97.5% quantile = 2.0086

Quantiles for normal distribution:

Normal Distribution: 2.5% quantile = -1.9600, 97.5% quantile = 1.9600
```

Confidence Interval for a Proportion

- $X_1, ..., X_n \sim Ber(p)$.
- We want a confidence interval for p.
- Taking $\widehat{p} = \overline{X}_n$. Because it's a type of a mean we know that:
  1. $E[\widehat{p}] = p$.
  2. $Var(\widehat{p}) = \frac{p(1-p)}{n}$.
  3. As n increases $\hat{p}$ approximately has a $N(p, \frac{p(1-p)}{n})$ distribution.
- Therefore, it can be shown that

$$CI = \widehat{p} \pm z_{1-\alpha/2} \cdot \sqrt{\frac{\widehat{p}(1-\widehat{p})}{n}}.$$

is a valid confidence interval at confidence level $1-\alpha$.

Tips Example: Confidence Interval for a Proportion

```
# 95% Confidence interval for the probablity of tipping more than 5
alpha = 0.05
z = norm.ppf(1 - alpha / 2)
n = tips.shape[0]
p_hat = (tips["tip"] > 5).mean()
SE_p = (p_hat * (1 - p_hat) / n) ** 0.5
CI = (p_hat - z * SE_p, p_hat + z * SE_p)
print('95% confidence interval for the probability of tipping more than $5: ',
(round(CI[0], 3), round(CI[1], 3)))

# Now for the probability of smoker
p_hat = (tips["smoker"] == 'Yes').mean()
SE_p = (p_hat * (1 - p_hat) / n) ** 0.5
CI = (p_hat - z * SE_p, p_hat + z * SE_p)
print('95% confidence interval for the probability of being a smoker: ',
(round(CI[0], 3), round(CI[1], 3)))

# Now for the probability of being a smoker given that the tip is more than 5
p_hat = (tips[tips["tip"] > 5]["smoker"] == 'Yes').mean()
SE_p = (p_hat * (1 - p_hat) / n) ** 0.5
CI = (p_hat - z * SE_p, p_hat + z * SE_p)
print('95% confidence interval for the probability of being a smoker given that the tip is more than $5:
(round(CI[0], 3), round(CI[1], 3)))
```

```
95% confidence interval for the probability of tipping more than $5:  (0.041, 0.107)
95% confidence interval for the probability of being a smoker:  (0.32, 0.442)
95% confidence interval for the probability of being a smoker given that the tip is more than $5:  (0.274, 0.392)
```

Choose Sample Size For CIs

$$CI = \bar{X} \pm z_{1-\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$$

- The length of the confidence interval, denoted by W, is

$$W = 2z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}$$

- If σ is indeed known, we can design our study so n is chosen to get a confidence interval of a certain width W by solving for n.
- If σ is replaced with S (and t distribution) or in the case of the proportion, the length of the confidence interval is *random*. But we can still evaluate this for different initial guesses.

More confidence intervals

- When the asymptotic distribution of our estimator is normal, as is the case with MLE and MoM estimators, we can have a general 'recipe' for confidence intervals. If $\hat{\theta} \sim N(\theta, V_\theta)$ then a CI for θ is

$$CI = \hat{\theta} \pm z_{1-\alpha/2} \hat{V}_\theta$$

where $\hat{V}_\theta$ is estimated variance. The way to estimate the variance is not always straightforward.

More confidence intervals

- Not all confidence intervals are symmetric.
- Let $X_1, X_2, \ldots, X_n$ be i.i.d. from $\mathcal{N}(\mu, \sigma^2)$.
- A confidence interval for σ^2 is

$$\left(\frac{(n-1)S^2}{\chi^2_{1-\alpha/2,n-1}}, \frac{(n-1)S^2}{\chi^2_{\alpha/2,n-1}} \right).$$

(χ^2_{n-1} is another distribution we haven't discussed)

Introduction to Monte Carlo Simulations

- The **Monte Carlo Method** is a computational technique that uses random sampling to estimate mathematical or statistical quantities.
- The general ideas is that we can:
  1. Simulate data from a known distribution or process.
  2. Use the simulated data to compute estimates or solve problems.

Introduction to Monte Carlo Simulations

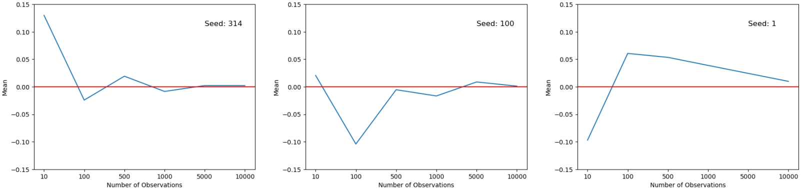
- The **Monte Carlo Method** is a computational technique that uses random sampling to estimate mathematical or statistical quantities.
- The general ideas is that we can:
  1. Simulate data from a known distribution or process.
  2. Use the simulated data to compute estimates or solve problems.
- Some applications:
  - Estimating probabilities.
  - Evaluating expectations (and more generally integrals)
  - Understanding the properties of estimators (e.g., bias, variance).

MC & LLN

- Why does it work ?

MC & LLN

- Why does it work ?
- Law of Large Numbers says that the mean approaches the expectation. So we can use the computer to "sample" as many as data points that we want.
- Below, I sampled 10,000 observations $N(0, 1)$, and calculated the mean of the first 10,100,500,1,000,5,000, and 10,000 observations.



MC Example (1): PMF

- Starting with a probability example.
- Example (314): A fair coin is flipped 20 times. Let X be the longest streak of any type.
- Calculate $\Pr(X = 4)$.
- **The hard way:** go over each of the 2^{20} possible outcomes (runs of HT) and for each result check if the longest streak is exactly 4.
- **The simple way:** simulate vectors from this distribution, and look at the empirical proportion of vectors with longest streak being 4.
- Now repeat for $\Pr(X = x)$.

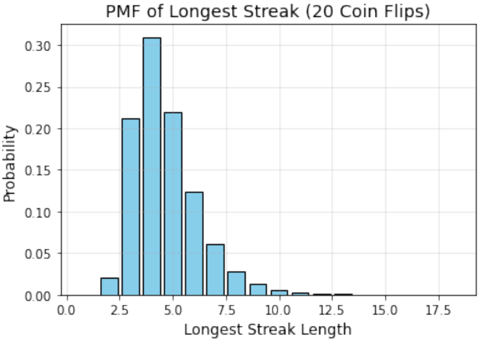
MC Example (1): PMF

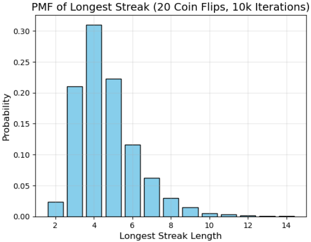
- In this case, the steps are as follows.
- Define the simulation parameters.
  - 20 flips
  - Number of simulation iterations (here 100,000)
- Write a function that given a sample, counts the longest streak.
- Simulate data "number of iterations" times, and at each time calculate the longest streak.
- Calculate the empirical $\widehat{Pr}(X = x)$ from the simulations.

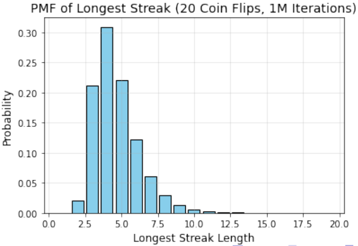
MC Example (1): PMF

```
import numpy as np
from collections import Counter
import matplotlib.pyplot as plt

# Parameters
n_flips = 20  # Number of flips
n_simulations = 100000  # Number of Monte Carlo simulations

# Function to compute the longest streak in a binary sequence
def longest_streak(sequence):
   max_streak = current_streak = 1
    for i in range(1, len(sequence)):
      if sequence[i] == sequence[i - 1]:
         current_streak += 1
              else:
         current_streak = 1
      max_streak = max(max_streak, current_streak)
            return max_streak

# Simulate longest streaks
longest_streaks = []
for _ in range(n_simulations):
    sequence = np.random.choice([0, 1], size=n_flips)  # 0 for tails, 1 for heads
    longest_streaks.append(longest_streak(sequence))

# Calculate the PMF
streak_counts = Counter(longest_streaks)
pmf = {k: v / n_simulations for k, v in streak_counts.items()}
```

MC Example (1): PMF







MC Example (2): Delivery Time Distribution

- A logistics company wants to estimate the average delivery time for packages in a city.
- Delivery time depends on:
  - **Travel Distance** (D): Modeled as $D \sim U(10, 50)$ (km).
  - **Warehouse Loading Time** (W): Modeled as $W \sim \text{Exponential}(0.2)$ (hours).
  - **Travel Speed** (S): Modeled as $S \sim U(20, 60)$ (km/h).
- Estimate the **expected delivery time** $E[T]$ using Monte Carlo simulations, where:

$$T = \frac{D}{S} + W.$$

MC Example (2): Delivery Time Distribution

1. Generate n_{sim} random samples for D, W, and S:
   - $D \sim U(10, 50)$.
   - $W \sim Exp(0.2)$.
   - $S \sim U(20, 60)$.
2. Compute delivery time for each simulation: $T_i = \frac{D_i}{S_i} + W_i$
3. Estimate the mean delivery time: $\widehat{E}[T] = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} T_i$.
4. We can also estimate probabilities, or the entire distribution.

MC Example (2): Delivery Time Distribution

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
n_sim = 10000  # Number of Monte Carlo simulations
lambda_W = 0.2  # Rate parameter for exponential loading time
distance_min, distance_max = 10, 50  # Uniform range for distance (km)
speed_min, speed_max = 20, 60  # Uniform range for speed (km/h)

# Simulate random inputs
distance = np.random.uniform(distance_min, distance_max, n_sim)
loading_time = np.random.exponential(1 / lambda_W, n_sim)
speed = np.random.uniform(speed_min, speed_max, n_sim)

# Compute delivery time
delivery_time = distance / speed + loading_time

# Estimate the mean delivery time
mean_delivery_time = np.mean(delivery_time)
print(f"Estimated Mean Delivery Time: {mean_delivery_time:.2f} hours")

# Estimate the probability of delivery time less than 2 hours
prob_delivery_time_lt_2 = np.mean(delivery_time < 2)
print(f"Estimated Probability of Delivery Time < 2 hours: {prob_delivery_time_lt_2:.2f}")

# Estimate the probability of delivery time more than 24 hours
prob_delivery_time_gt_24 = np.mean(delivery_time > 24)
print(f"Estimated Probability of Delivery Time > 24 hours: {prob_delivery_time_gt_24:.2f}")
```

MC Example (2): Delivery Time Distribution

```
Estimated Mean Delivery Time: 5.91 hours
Estimated Probability of Delivery Time < 2 hours: 0.20
Estimated Probability of Delivery Time > 24 hours: 0.01
```

MC Example (2): Delivery Time Distribution

```
# Estimate the standard deviation of delivery time
std_delivery_time = np.std(delivery_time)
print(f"Estimated Standard Deviation of Delivery Time: {std_delivery_time:.2f}

# Estimate the mean log delivery time
mean_log_delivery_time = np.mean(np.log(delivery_time))
print(f"Estimated Mean Log Delivery Time: {mean_log_delivery_time:.2f}")
```

```
Estimated Standard Deviation of Delivery Time: 5.01 hours
Estimated Mean Log Delivery Time: 1.44
```

MC Example (2): Delivery Time Distribution

```
# Estimate the standard deviation of delivery time
std_delivery_time = np.std(delivery_time)
print(f"Estimated Standard Deviation of Delivery Time: {std_delivery_time:.2f}

# Estimate the mean log delivery time
mean_log_delivery_time = np.mean(np.log(delivery_time))
print(f"Estimated Mean Log Delivery Time: {mean_log_delivery_time:.2f}")
```

```
Estimated Standard Deviation of Delivery Time: 5.01 hours
Estimated Mean Log Delivery Time: 1.44
```

- We can repeat the process for different scenarios: what if we had more drivers, or we would have increased the limit distance.

MC example (3): Study Estimators

- Let $X_1, ..., X_n \sim Exp(\lambda)$.
- It can be shown that the MLE is $\widehat{\lambda}_{MLE} = \frac{1}{\bar{X}_n}$.

MC example (3): Study Estimators

- Let $X_1, ..., X_n \sim Exp(\lambda)$.
- It can be shown that the MLE is $\widehat{\lambda}_{MLE} = \frac{1}{\bar{X}_n}$.
- An alternative estimator was proposed based on the median.
  - Because $\Pr(X < x) = 1 - e^{-\lambda x}$, then for the median we have $0.5 = 1 - e^{-\lambda q_{0.5}}$. Solving for the median we get $\lambda = \frac{ln(2)}{q_{0.5}}$.
- Therefore, the proposal is

$$\widehat{\lambda}_{Med} = \frac{ln(2)}{X_{med}}$$

where X_{med} is the sample median.

MC example (3): Study Estimators

- Let $X_1, ..., X_n \sim Exp(\lambda)$.
- It can be shown that the MLE is $\widehat{\lambda}_{MLE} = \frac{1}{\bar{X}_n}$.
- An alternative estimator was proposed based on the median.
  - Because $\Pr(X < x) = 1 - e^{-\lambda x}$, then for the median we have $0.5 = 1 - e^{-\lambda q_{0.5}}$. Solving for the median we get $\lambda = \frac{ln(2)}{q_{0.5}}$.
- Therefore, the proposal is

$$\widehat{\lambda}_{Med} = \frac{ln(2)}{X_{med}}$$

  where X_{med} is the sample median.
- Which estimator is better? are they biased? consistent?

MC example (3): Study Estimators

- Process is as follows. Say for $n = 30$, and start with the case $\lambda = 2$
- Create n_{sim} *datasets* (n_{sim} is large), each with 30 observations from the $Exp(2)$ distribution.
- At each data set, calculate $\widehat{\lambda}_{MLE}$ and $\widehat{\lambda}_{Med}$.
- By LLN, the means across datasets of $\widehat{\lambda}_{MLE}$ and $\widehat{\lambda}_{Med}$ are good approximations of the expectations $E[\widehat{\lambda}_{MLE}]$ and $\widehat{\lambda}_{Med}$.
- Calculate the bias $E[\widehat{\lambda}_{MLE}] - \lambda$ and $E[\widehat{\lambda}_{Med}] - \lambda$
- Similarly "estimate" the variance and the MSE.
- We can repeat for different n and different λ.

MC example (3): Study Estimators

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Parameters
n_sim = 10000  # Number of Monte Carlo simulations
n = 30  # Sample size
true_lambda = 2  # True rate parameter for Exponential distribution

# Functions to compute estimators
def mle_estimator(sample):
return 1 / np.mean(sample)

def median_estimator(sample):
return np.log(2) / np.median(sample)

# Storage for results
mle_estimates = []
median_estimates = []

# Monte Carlo simulation
np.random.seed(314)
for _ in range(n_sim):
sample = np.random.exponential(scale=1 / true_lambda, size=n)
mle_estimates.append(mle_estimator(sample))
median_estimates.append(median_estimator(sample))

# Convert to arrays
mle_estimates = np.array(mle_estimates)
median_estimates = np.array(median_estimates)
```

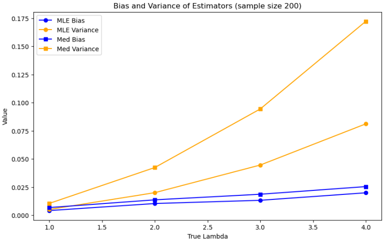
MC example (3): Study Estimators

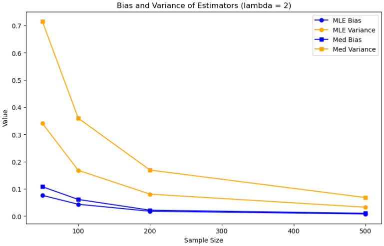
```
# Compute metrics
def compute_metrics(estimates, true_value):
bias = np.mean(estimates) - true_value
variance = np.var(estimates)
mse = bias**2 + variance
return bias, variance, mse

mle_bias, mle_variance, mle_mse = compute_metrics(mle_estimates, true_lambda)
median_bias, median_variance, median_mse = compute_metrics(median_estimates, true_lambda)

# Print results
print("MLE:")
print(f"  Bias: {mle_bias:.4f}, Variance: {mle_variance:.4f}, MSE: {mle_mse:.4f}")
print("Median-Based Estimator:")
print(f"  Bias: {median_bias:.4f}, Variance: {median_variance:.4f}, MSE: {median_mse:.4f}")
```

```
MLE:
  Bias: 0.0671, Variance: 0.1492, MSE: 0.1537
Median-Based Estimator:
  Bias: 0.0813, Variance: 0.3167, MSE: 0.3234
```

MC example (3): Study Estimators

- A more rigorous study: compare different sample sizes n and true parameter λ values.





MC example (3): Study Coverage of CIs

- **Goal:** Evaluate the coverage rate of confidence intervals for the mean of a normal distribution.
- Here, we will consider the case the data is normal, sample size is small ($n = 20$) and we will compare between the CI using t distribution and the normal distribution.
- For each simulation iteration:
  1. Create a sample size from the normal distribution.
  2. Calculate the two confidence intervals.
  3. Check if the CIs include the parameter.
- Then, calculate the *empirical proportion* of CIs that includes the true parameter μ.

MC example (3): Study Coverage of CIs

```
# Parameters
n_sim = 100000  # Number of Monte Carlo simulations
n = 20  # Sample size
true_mu = 10  # True mean
true_sigma = 3  # True standard deviation
alpha = 0.05  # Significance level for confidence intervals

# Storage for coverage results
coverage_t = 0
coverage_z = 0

# Monte Carlo simulation
np.random.seed(314)
for _ in range(n_sim):
# Generate a sample from the normal distribution
sample = np.random.normal(loc=true_mu, scale=true_sigma, size=n)
sample_mean = np.mean(sample)
sample_std = np.std(sample, ddof=1)  # Sample standard deviation

# T-distribution confidence interval
t_critical = stats.t.ppf(1 - alpha / 2, df=n - 1)
lower_t = sample_mean - t_critical * (sample_std / np.sqrt(n))
upper_t = sample_mean + t_critical * (sample_std / np.sqrt(n))
if lower_t <= true_mu <= upper_t:
coverage_t += 1

# Z-distribution confidence interval (asymptotic normality)
z_critical = stats.norm.ppf(1 - alpha / 2)
lower_z = sample_mean - z_critical * (sample_std / np.sqrt(n))
upper_z = sample_mean + z_critical * (sample_std / np.sqrt(n))
if lower_z <= true_mu <= upper_z:
coverage_z += 1
```

MC example (3): Study Coverage of CIs

```
# Calculate coverage rates
coverage_rate_t = coverage_t / n_sim
coverage_rate_z = coverage_z / n_sim

# Print results
print(f"Coverage Rate (T-distribution): {coverage_rate_t:.4f}")
print(f"Coverage Rate (Z-distribution): {coverage_rate_z:.4f}")
```

```
Coverage Rate (T-distribution): 0.9490
Coverage Rate (Z-distribution): 0.9345
```

MC example (3): Study Coverage of CIs

Repeat it with the following changes (data are no longer normal)

```
true_lambda = 0.5  # True rate parameter
  sample = np.random.exponential(scale=1 / true_lambda, size=n)
```

Everything else stays the same

```
Coverage Rate (T-distribution): 0.9196
Coverage Rate (Z-distribution): 0.9054
```

Now with larger sample size

```
true_lambda = 0.5  # True rate parameter
sample = np.random.exponential(scale=1 / true_lambda, size=n)
```

```
Coverage Rate (T-distribution): 0.9430
Coverage Rate (Z-distribution): 0.9401
```