# Project No2 - Concentration Theorems

When writing code, we recommend you to be as modular as possible. For example, if you are running multiple experiments for different choices of parameters, it may be convenient to write a function that does one experiment and then make multiple calls to the same function. Follow the *do not repeat yourself* rule when writing code!

In [11]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
import random
import math

import matplotlib as mpl

mpl.rcParams['font.size'] = 16
```

## 0. Setup

We will consider $X_1, X_2, \dots, X_n$ a sequence of i.i.d. random variables with mean $\mathbb{E}[X_1] = \mu$. We define the sample mean as

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

The objective of this project is to study the behavior of $\bar{X}_n$ as $n \to \infty$.

## 1. Law of Large Numbers

Here we are going to focus in the convergence in probability and almost surely of the sample mean of i.i.d. random variables. Remember that a sequence of random variables $Y_1, Y_2, \dots$ converges in probability to a random variable $Y$ if for every $\epsilon > 0$ we have

$$\lim_{n\to\infty} \mathbb{P}\left(|Y_n - Y| > \epsilon\right) = 0$$

The **weak law of large numbers** states

> The sample mean $\bar{X}_n$ converges in probability to $\mu$.

The **strong law of large numbers** states a more strict sense in which this convergence happens

> The sample mean $\bar{X}_n$ converges almost surely to $\mu$.

Let's see how these two behave numerically.

### 1.1. Gaussian case

Consider the simple case where $X_i \sim N(0, 1)$. Compute the sample mean for increasing values of $n$ and show how the sample mean converges to $0$. To do this, you can simply show a plot of the sample mean as a function of $n$.

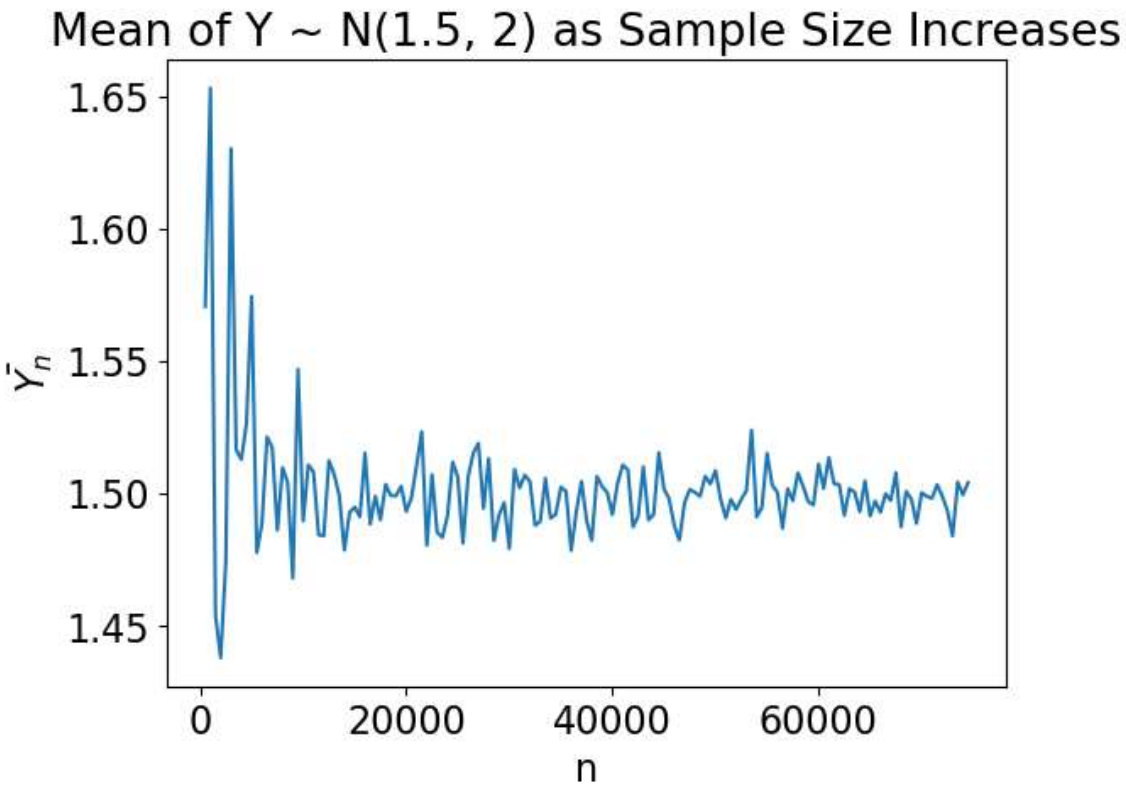
***Tip:*** You can compute the sample mean of a normal random variable by simply doing

```
n = 100
one_sample_mean = np.mean(np.random.normal(loc=0.0, scale=1.0, siz
e=n))
```

The same works if you want to compute a total of `n_sim` sample means:

```
n = 100
n_sim = 500
many_sample_means = np.mean(np.random.normal(loc=0.0, scale=1.0, s
ize=(n, n_sim)), axis=0)
```

In [20]:
```
np.random.seed(20231207)
sample_sizes = np.arange(500, 75_000, 500)
sample_means = np.zeros(len(sample_sizes))

for i, size in enumerate(sample_sizes):
    data = np.random.normal(loc=1.5, scale=2.0, size=size)
    sample_mean = np.mean(data)
    sample_means[i] = sample_mean


plt.plot(sample_sizes, sample_means)

plt.ylabel(r'$\bar{Y_n}$')
plt.xlabel('n')
plt.title('Mean of Y ~ N(1.5, 2) as Sample Size Increases')

plt.show()
```




In [ ]:

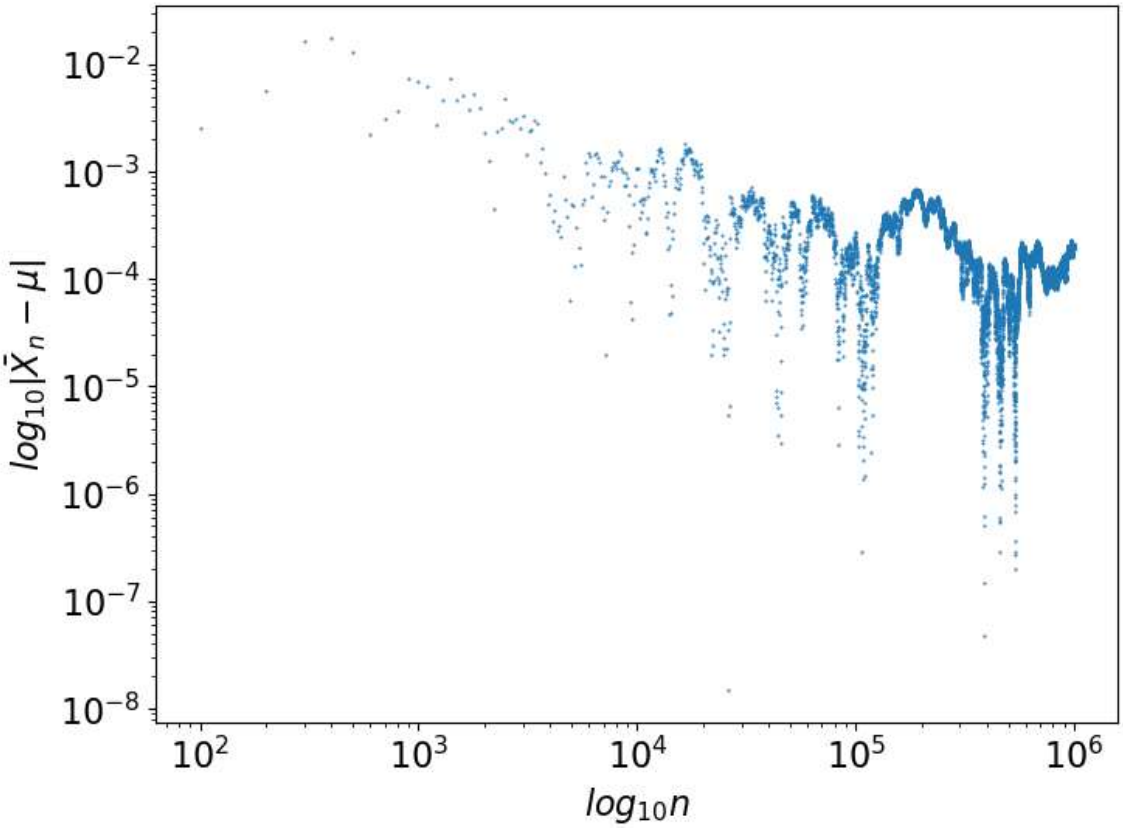
In [ ]:

### 1.2. Beta distribution

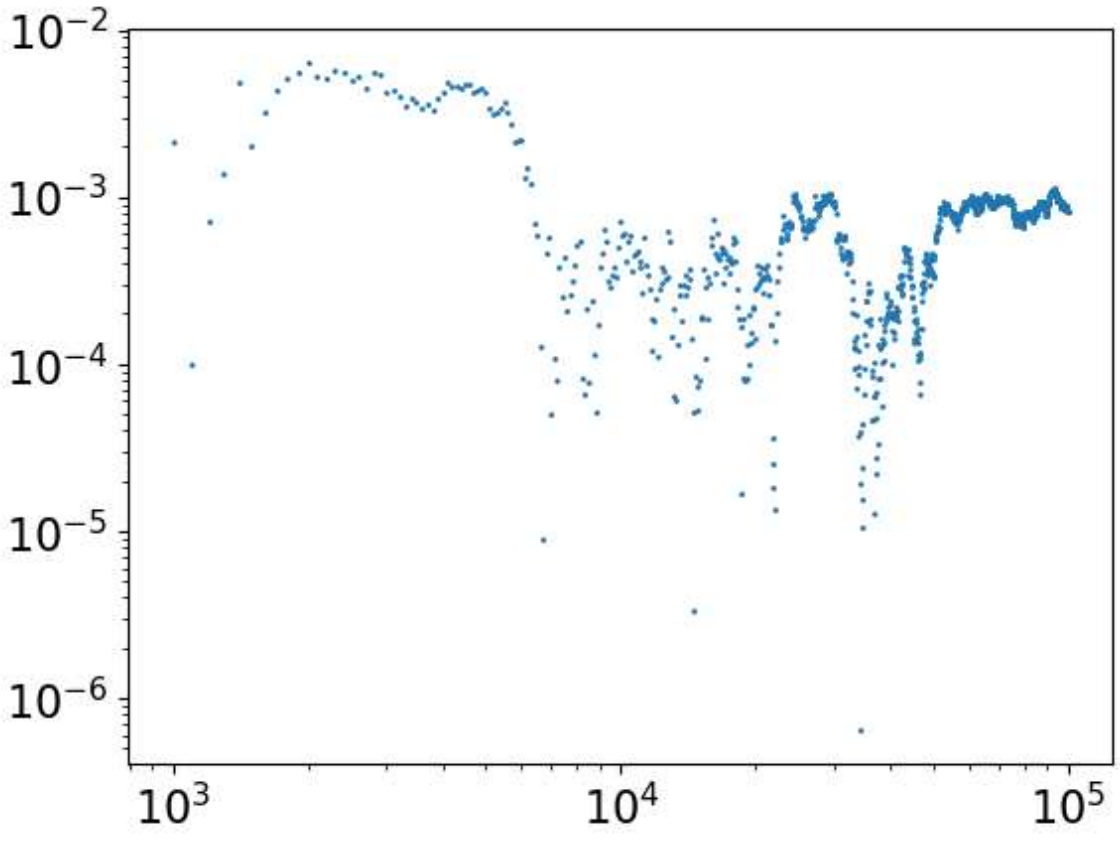
Now, repeat the same experiment but for a Beta distribution with parameters of your choice. Remember that the mean of a Beta distribution with parameters $a$ and $b$ is $a/(a+b)$.

For this section, it is important that you compute the sample mean as you increase the value of *n for the same realization of the random variables* $X_i$. To do this, first sample all the values of $X_1, X_2, \ldots, X_{n_{max}}$ just one time and then compute the partial averages $\bar{X}_n$ for different values of $n \leq n_{max}$.

Make this plot is logarithmic scale for both axes. This is how your solution should look like. If



In [21]:
```
param_alpha = 3
param_beta = 1
param_mu = param_alpha / (param_alpha + param_beta)

beta_means_list = []

n_betas = np.random.beta(param_alpha, param_beta, size=100000)

sample_sizes = np.arange(1000, 100000, 100)

for size in sample_sizes:
    beta_means_list.append(abs(np.mean(n_betas[0:size]) - param_mu))

log_sample_sizes = np.log(sample_sizes)

plt.scatter(sample_sizes, beta_means_list, s=1)
plt.xscale('log')
plt.yscale('log')

plt.show()
```




In [ ]:

In [ ]:

In [ ]:

**To Discuss:** When doing this calculations, what time of convergence are we studying? almost surely or in probability?

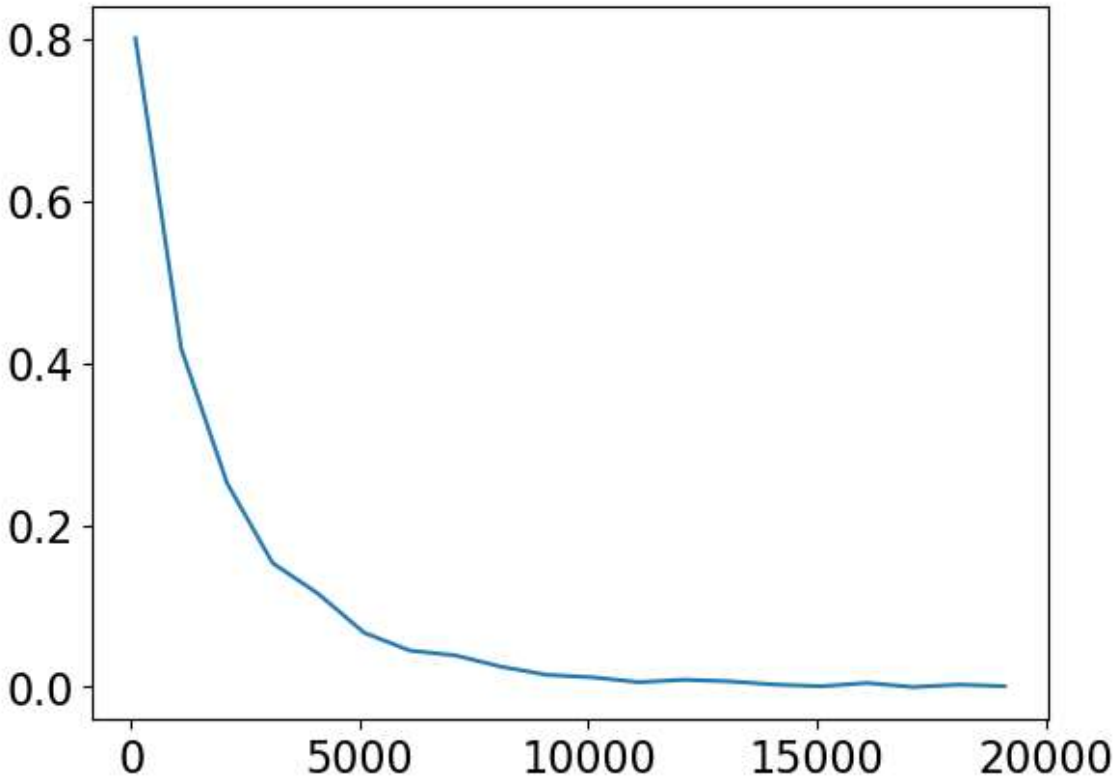
**To Discuss:** Can you identify the general trend in the previous plot? Can you give meaning the low peaks in the plot?

### 1.3. Compute the probability

Now, instead of evaluating the value of $\bar{X}_n$, compute the probability $\mathbb{P}\left(|\bar{X}_n| > \epsilon\right)$ for different values of $n$ and $\epsilon$ and plot the result as a function of $n$. What do you observe? See the next item for an example of the solution for this problem.

In [14]:
```
new_epsilon = 0.025
result_list = []
sample_sizes_list = np.arange(100, 20000, 1000)

for new_N in sample_sizes_list:
    game_list = []
    for new_repetition in range(0, 1000):
        new_sample_mean = np.mean(np.random.normal(loc=0.0, scale=1.0, size
        if abs(new_sample_mean) >= new_epsilon:
            game_list.append(1)
        else:
            game_list.append(0)
    result_list.append((sum(game_list) / len(game_list)))

plt.plot(sample_sizes_list, result_list)
```

Out[14]:
```
[<matplotlib.lines.Line2D at 0x7fe2b46f73d0>]
```




In [ ]:

In [ ]:

In [ ]:

### 1.4. Concentration bounds

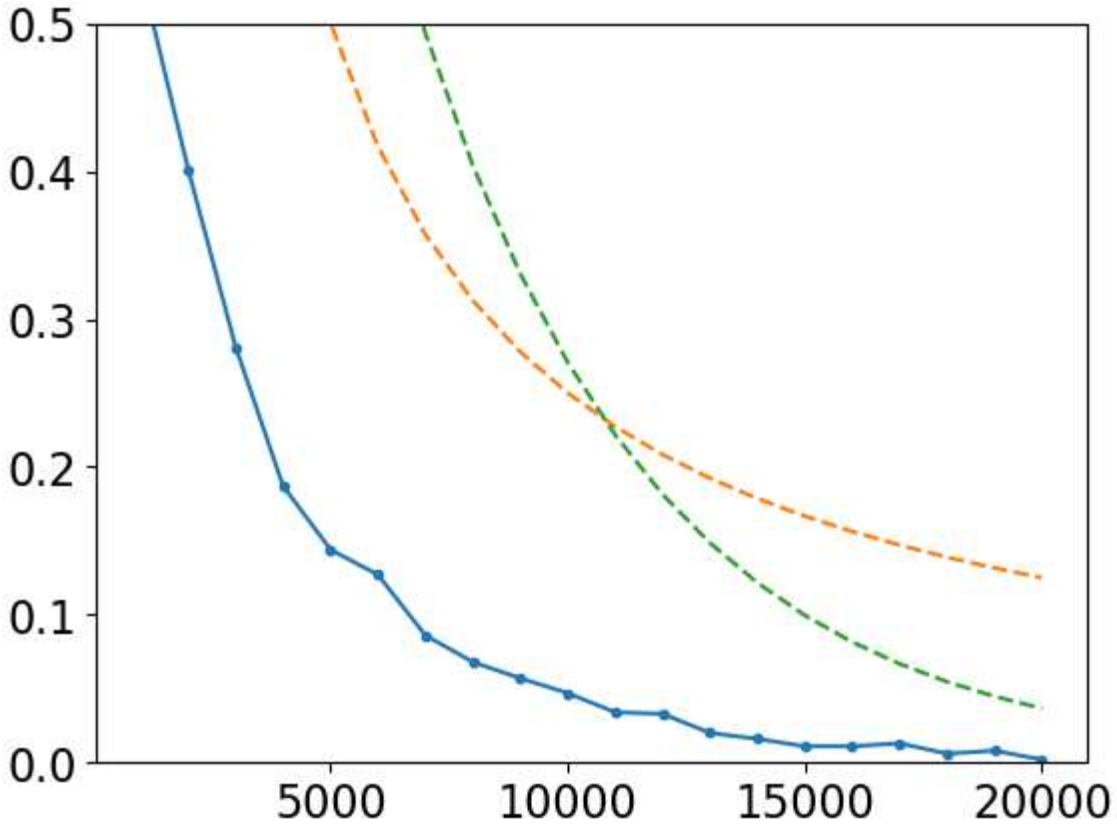
Repeat the same plot that in the last item and show how it compares with the upper bound obtain from using Chebyshev's inequality

$$\mathbb{P}\left(|\bar{X}_n| > \epsilon\right) \leq \frac{1}{n\epsilon^2}$$

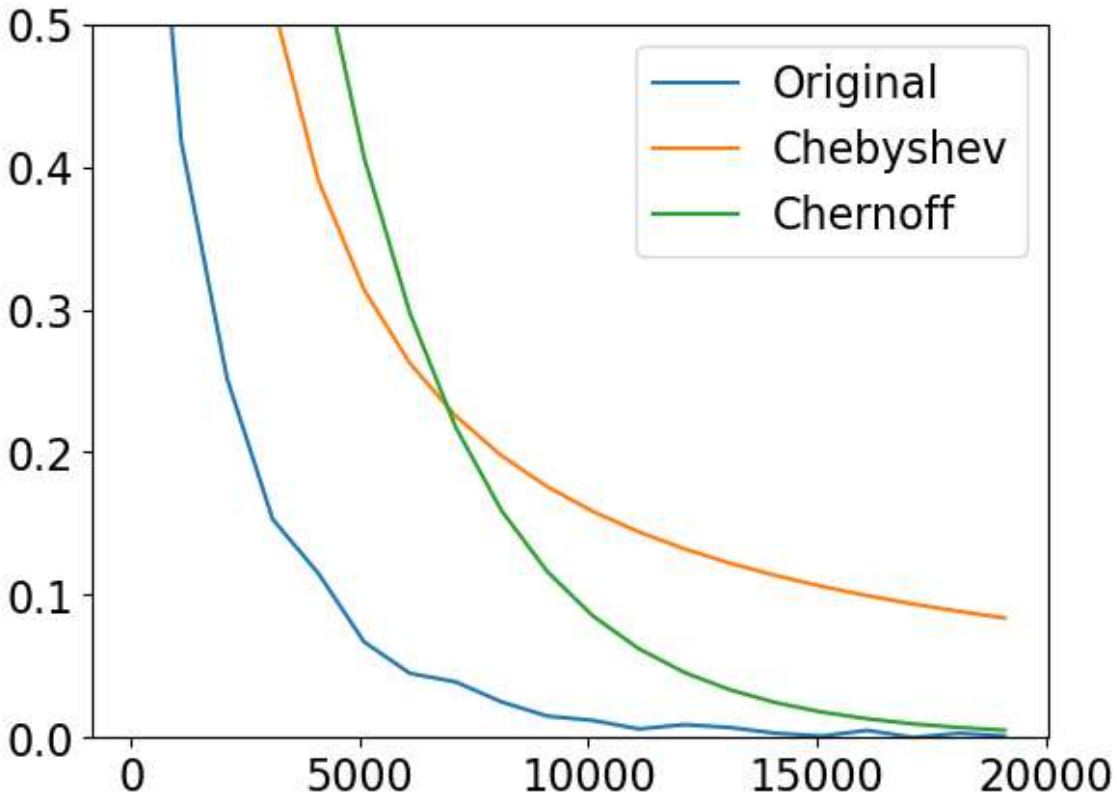
and the Chernoff bound

$$\mathbb{P}\left(|\bar{X}_n| > \epsilon\right) \leq 2\exp\left(-\frac{\epsilon^2 n}{2}\right)$$

At the end of the day, you should obtain something like this



In [15]:
```
new_chebyshev = []
new_chernoff = []

for new_N in sample_sizes_list:
    new_chebyshev.append(1 / (new_N * (new_epsilon ** 2)))
    new_chernoff.append(2 * np.exp((-new_N * (new_epsilon ** 2)) / 2))

plt.plot(sample_sizes_list, result_list, label='Original')
plt.plot(sample_sizes_list, new_chebyshev, label='Chebyshev')
plt.plot(sample_sizes_list, new_chernoff, label='Chernoff')
plt.ylim((0, 0.5))
plt.legend()
plt.show()
```




In [ ]:

In [ ]:

In [ ]:

## 2. Central Limit theorem

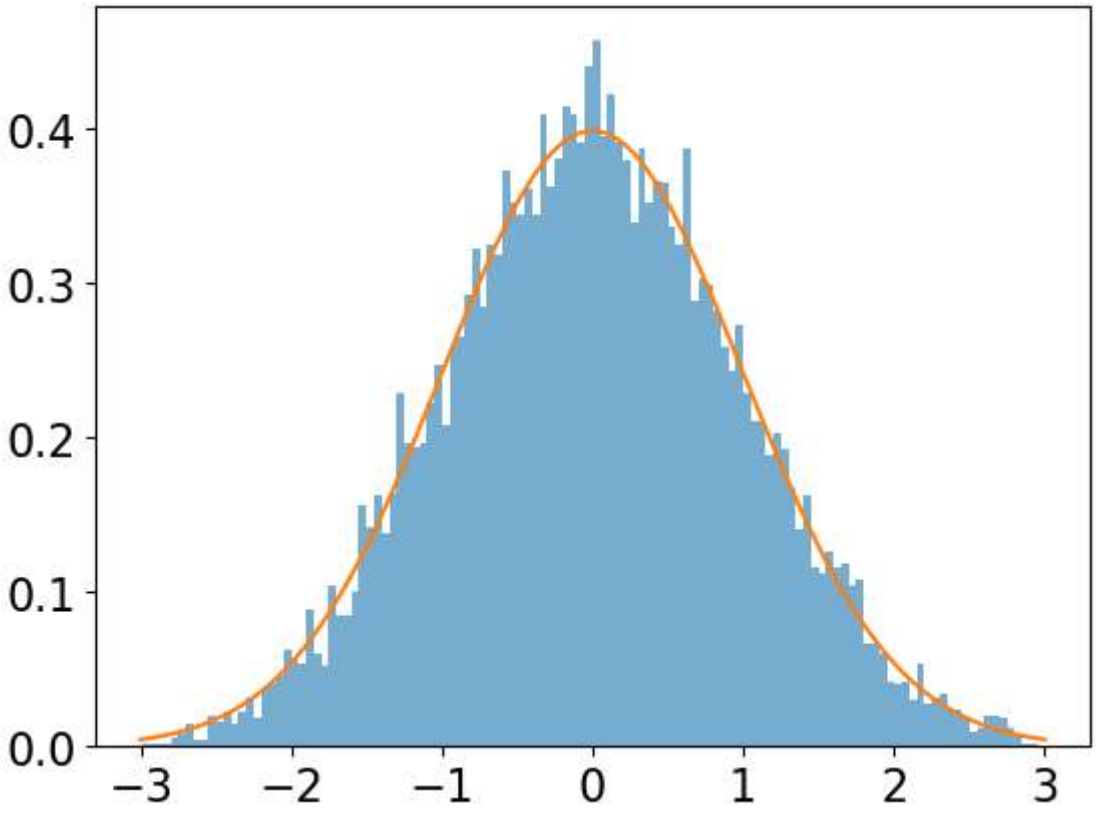
Let's first state one more time the Central Limit Theorem

> Consider $X_1, X_2, \ldots, X_n$ a sequence of i.i.d. random variables with mean $\mathbb{E}[X_1] = \mu$ and finite variance $\mathbb{V}ar[X_1] = \sigma^2$. If call
>
> $$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$
>
> the sample mean, then
>
> $$\sqrt{n}\frac{\bar{X}_n - \mu}{\sigma}$$
>
> converges in distribution to the standard Normal distribution $N(0, 1)$.

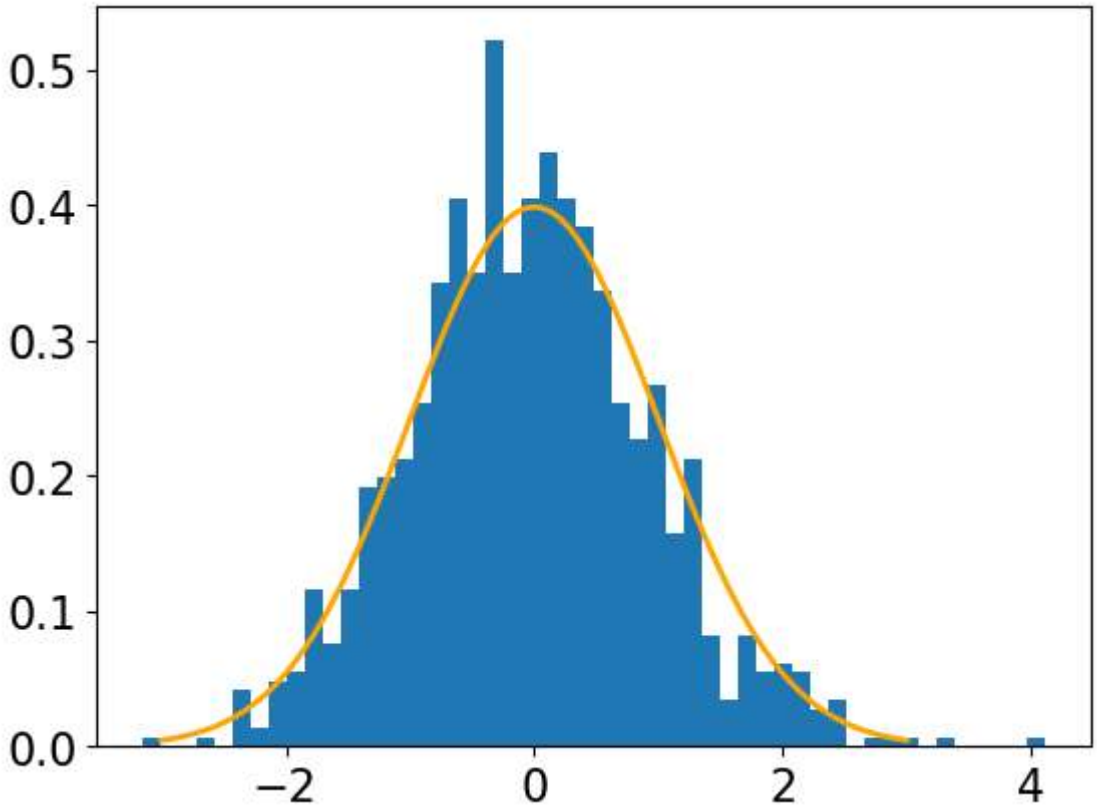
### 2.1. Continuous case

Pick a continuous random variable with bounded second moment and show that the central limit theorem holds numerically. Sample the sample mean $\bar{X}_n$ for various numbers of $n$ and show that $\sqrt{n}(\bar{X}_n - \mu)/\sigma$ converges to the standard normal distribution.

This is how the solution should look like for the mean of Beta distributions and a total of 10000 simulations.



In [17]:
```
num_samples = int(1e3)
shape_param = 1
scale_param = 1
gamma_mean = shape_param * scale_param
gamma_sd = np.sqrt(shape_param) * scale_param
sample_sizes = np.arange(1, num_samples + 1)

sample_means = [np.average(np.random.gamma(shape=shape_param, scale=scale_p
z_scores = np.sqrt(sample_sizes) * (np.array(sample_means) - gamma_mean) /

plt.hist(z_scores, density=True, bins=50)
x_values = np.linspace(-3, 3, 100)
pdf_standard_normal = stats.norm.pdf(x_values, 0, 1)
plt.plot(x_values, pdf_standard_normal, linewidth=2, color="orange")
plt.show()
```

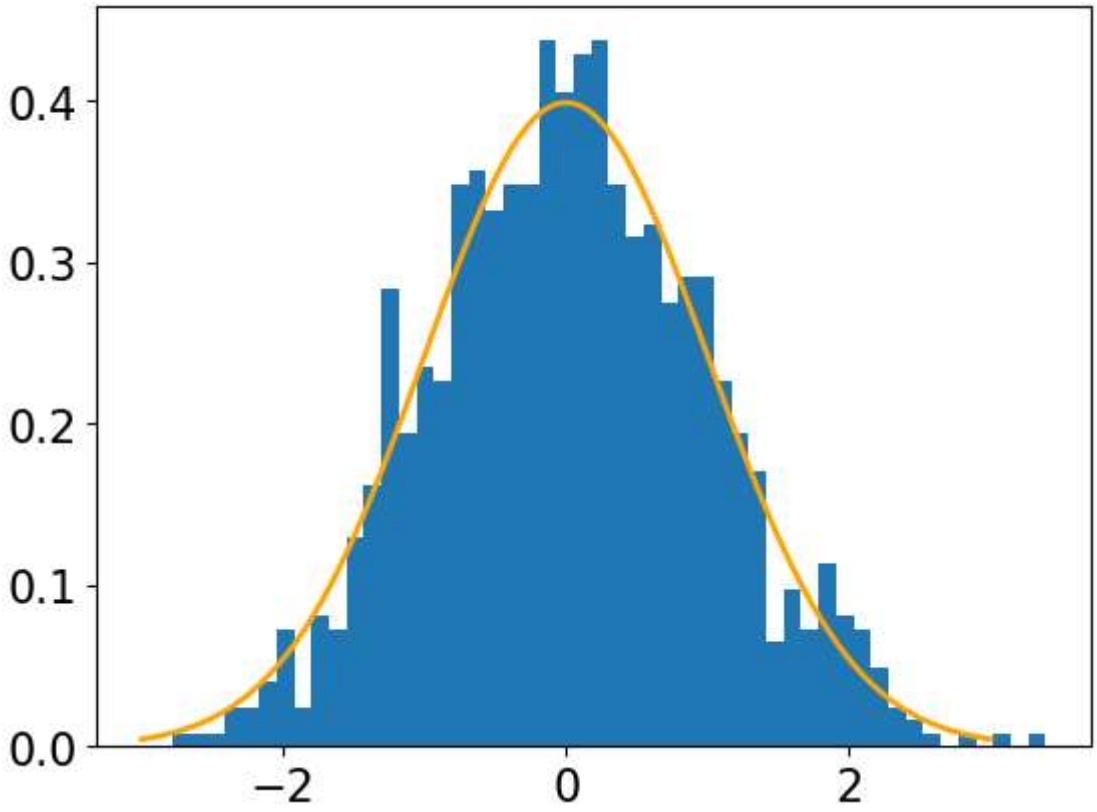



In [ ]:
```
```

### 2.2. Discrete case

Repeat the experiment but now with a discrete random variable. When plotting the histogram, be careful on how you define the bins.

In [18]:
```
num_samples = int(1e3)
num_bins = 20
prob_binomial = 0.7
binomial_mean = num_bins * prob_binomial
binomial_sd = np.sqrt(num_bins * prob_binomial * (1 - prob_binomial))
sample_sizes = np.arange(1, num_samples + 1)

sample_means = [np.average(np.random.binomial(num_bins, prob_binomial, size
z_scores = np.sqrt(sample_sizes) * (np.array(sample_means) - binomial_mean)

plt.hist(z_scores, density=True, bins=50)
x_values = np.linspace(-3, 3, 100)
pdf_standard_normal = stats.norm.pdf(x_values, 0, 1)
plt.plot(x_values, pdf_standard_normal, linewidth=2, color='orange')
plt.show()
```




### 2.3. [Optional] Distribution without second moment

Consider now a distribution with defined mean without finite second moment. In principle, the central limit theorem does not apply for this case. Generate a simulation of the sample mean for a distribution with these properties and try to understand the behavior of the scaled sample mean as $n$ increases. You may find interesting the reading about [stable distributions (https://en.wikipedia.org/wiki/Stable_distribution)](https://en.wikipedia.org/wiki/Stable_distribution).

## ### Contribution statement

Please include a list of the students you work with along this project (including yourself). Include both names and GitHub usernames of the people you collaborated with. Maximum of three persons per group.

- Member No1: Cherno Barry
- Member No2:
- Member No3: