Lab Session - 1 : Asymptotics and simulation

MATH350 – Statistical Inference

Dr. Tanujit Chakraborty, Ph.D. from ISI Kolkata.
Assistant Professor in Statistics at Sorbonne University.
tanujit.chakraborty@sorbonne.ae

Course Webpage: https://www.ctanujit.org/SI.html

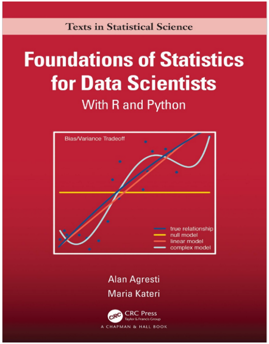
Course for BSc Mathematics and Data Science Students.

Asymptotic analysis

- Oftentimes, a very good approximate answer emerges when n is large (in other words, you have many samples). We call results that rely on this type of approximation as asymptotic.

- Computerized simulations can also be carried out to approximate sampling distributions.

- With a model we can draw many random samples, compute the statistic, and characterize it's sampling distribution.



Texts in Statistical Science

Foundations of Statistics for Data Scientists

With R and Python

Alan Agresti

Maria Kateri

CRC Press

Taylor & Francis Group

A CHAPMAN & HALL BOOK

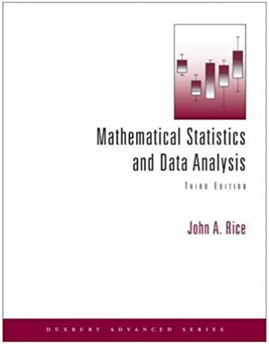
Figure: (Easy) Textbook

Asymptotic analysis

- Bootstrap is useful to compute the sampling distribution without a model! (to be discussed later)

- If we can just simulate, why do asymptotic analysis?

  1. Better understanding of the behavior. (Understanding the assumptions: What if X_i are not uniform? What if I don't really know the distribution of X_i ? Understanding the scaling: What if $n = 1000$ instead of 100? What if $n = 1,000,000$?)
  2. Faster to get an answer.



Mathematical Statistics and Data Analysis

Third Edition

John A. Rice

Duxbury Advanced Series

Figure: See Chapter 5

Sampling Distribution

In theory. we've discussed a few examples of how to determine the distribution of a statistic computed from data, assuming a certain probability model for the data.

For example, we have shown the following results: If $X_1, \dots, X_n \overset{IID}{\sim} \mathcal{N}(0,1)$*, then*

$$\bar{X} \sim \mathcal{N}\left(0, \frac{1}{n}\right),$$

$$X_1^2 + \ldots + X_n^2 \sim \chi_n^2.$$

Reality check

For many (seemingly simple) statistics, it's difficult to describe its PMF or PDF exactly. For example:

1. *Suppose* $X_1, \ldots, X_{100} \overset{IID}{\sim}$ *Uniform* $(-1, 1)$. *What is the distribution of* $\bar{X}$ *?*

2. *Suppose* $(X_1, \ldots, X_6) \sim \text{Multinomial}\left(500, \left(\frac{1}{6}, \ldots, \frac{1}{6}\right)\right)$. *What is the distribution of*

$$T = \left(\frac{X_1}{500} - \frac{1}{6}\right)^2 + \ldots + \left(\frac{X_6}{500} - \frac{1}{6}\right)^2 ?$$

For questions that we don't know how to answer exactly, we'll try to answer them approximately.

Sample mean of IID uniform

If we fully specify the distribution of data, then we can always simulate the distribution of any statistic:

```
nreps = 10000
sample.mean = numeric(nreps)
n = 100
for (i in 1:nreps) {
X = runif(n, min = −1, max = 1)
sample.mean [i] = mean(X)
}
hist (sample.mean)
```

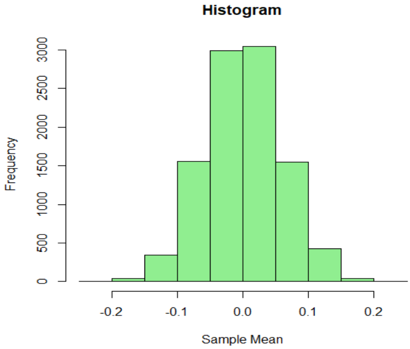


Figure: Histogram of sample.mean

Motivating Example : Dice Game

Is your friend cheating you in dice?

You are playing a dice game with your friend and he seems to be cheating (either that or you are really bad at this game). You deduce that the dice is not fair. This is, you expect each of the outcomes to be equally likely, but they do not seem to be coming up that way.

Approximating the multinomial distribution for large n

- Suppose $(Y_1, \ldots, Y_6) \sim \text{Multinomial}\left(n, \left(\frac{1}{6}, \ldots, \frac{1}{6}\right)\right)$. Y represents the number of times we obtain 1 through 6 when rolling a 6-sided die n times.

- For each $I = 1, \ldots, n$, let $\mathbf{X}^{(I)} = (1,0,0,0,0,0)$ if we got 1 on the I^{th} roll, $(0,1,0,0,0,0)$ if we got 2 on the I^{th} roll, etc. Then $(Y_1, \ldots, Y_6) = \mathbf{X}^{(1)} + \ldots + \mathbf{X}^{(n)}$.

- Let's apply the (multivariate) WLLN and CLT!

Approximating the multinomial distribution for large n

Let's write $\mathbf{X}^{(1)} = (X_1, \ldots, X_6)$, so $X_1, \ldots, X_6$ are random variables where exactly one them equals 1 (and the rest equal 0). Then:

$$\begin{aligned}
\mathbb{E}\left[X_i\right] = \mathbb{P}\left[X_i = 1\right] &= \frac{1}{6} \\
\operatorname{Var}\left[X_i\right] = \mathbb{E}\left[X_i^2\right] - \left(\mathbb{E}\left[X_i\right]\right)^2 &= \frac{1}{6} - \left(\frac{1}{6}\right)^2 = \frac{5}{36}, \\
\operatorname{Cov}\left[X_i, X_j\right] = \mathbb{E}\left[X_i X_j\right] - \mathbb{E}\left[X_i\right]\mathbb{E}\left[X_j\right] &= 0 - \left(\frac{1}{6}\right)^2 = -\frac{1}{36},
\end{aligned}$$

for $i \neq j$.

Approximating the multinomial distribution for large n

By the LLN, as $n \to \infty$,

$$\left(\frac{Y_1}{n}, \dots, \frac{Y_6}{n}\right) \to \left(\frac{1}{6}, \dots, \frac{1}{6}\right)$$

in probability. By the CLT, as $n \to \infty$,

$$\sqrt{n}\left(\frac{Y_1}{n} - \frac{1}{6}, \dots, \frac{Y_6}{n} - \frac{1}{6}\right) \to \mathcal{N}(0, \Sigma)$$

in distribution, where

$$\Sigma = \begin{pmatrix} \frac{5}{36} & -\frac{1}{36} & \cdots & -\frac{1}{36} \\ -\frac{1}{36} & \frac{5}{36} & \cdots & -\frac{1}{36} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{36} & -\frac{1}{36} & \cdots & \frac{5}{36} \end{pmatrix} \in \mathbb{R}^{6\times 6}$$

(The negative values of Σ_{ij} for $i \neq j$ mean Y_i and Y_j are, as expected, slightly anti-correlated.)

Is your friend cheating you in dice?

Recall

$$nT_n = n\left(\frac{Y_1}{n} - \frac{1}{6}\right)^2 + \ldots + n\left(\frac{Y_6}{n} - \frac{1}{6}\right)^2.$$

The function $g\left(x_1, \ldots, x_6\right) = x_1^2 + \ldots + x_6^2$ is continuous, so

$$nT_n \to Z_1^2 + \ldots + Z_6^2.$$

in distribution, where $(Z_1, \ldots, Z_6) \sim \mathcal{N}(0, \Sigma)$.

Hence, when n is large, the distribution of T_n is approximately that of $\frac{1}{n}\left(Z_1^2 + \ldots + Z_6^2\right)$.

Finally, what is the distribution of $Z_1^2 + \ldots + Z_6^2$?

Is your friend cheating you in dice?

Using bilinearity of covariance, it is easy to show that if

$$W_1, \ldots, W_6 \overset{IID}{\sim} \mathcal{N}(0,1),$$

then

$$\frac{1}{\sqrt{6}}\left(W_1 - \bar{W}, \ldots, W_6 - \bar{W}\right) \sim \mathcal{N}(0, \Sigma)$$

(Here $\bar{W} = \frac{1}{6}\left(W_1 + \ldots + W_6\right)$.) So $Z_1^2 + \ldots + Z_6^2$ has the same distribution as

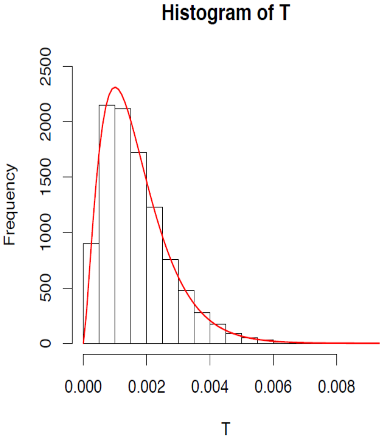
$$\frac{1}{6}\left(\left(W_1 - \bar{W}\right)^2 + \ldots + \left(W_6 - \bar{W}\right)^2\right)$$

This is the sample variance of 6 IID standard normals, which we have shown in theory class has distribution $\frac{1}{6}\chi_5^2$.

Conclusion: T_n has approximate distribution $\frac{1}{6n}\chi_5^2$.

Is your friend cheating you in dice?

- Here's our simulated histogram of T_n, overlaid with the (appropriately rescaled) PDF of the $\frac{1}{6n}\chi^2_5$ distribution:



R Code

```
nreps = 10000
T = numeric (nreps)
n = 500
p=c(1/6,1/6,1/6,1/6,1/6,1/6)
for (i in 1 : nreps){
X = rmultinom(1, n, p)
T[i]=sum($(X/n - p)^2$)
}
hist(T)
```

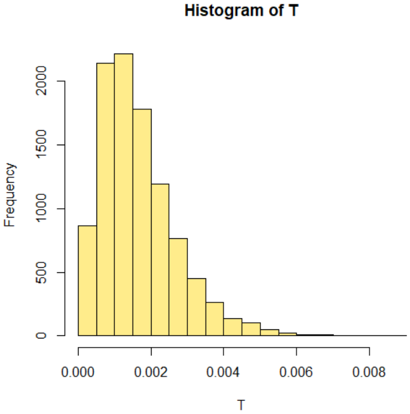


Figure: Histogram of sample.mean

EXAMPLES: SIMULATIONS IN R

Approximation through numerical simulation

Assume price $\sim$ Exponential (λ)

Consider samples of size $n = 201$

$E[\text{ price }] = \lambda^{-1}$ and $\text{Var}[\text{ price }] = \lambda^{-2}$ and therefore

$$\text{Var}(\overline{\text{ price }}) = \sqrt{(1/\lambda^2)\,/201} \approx 0.0705/\lambda$$

Recall that The Normal approximation for the sampling distribution of the

average price suggests

$$1/\lambda \pm 1.96 \cdot 1/(\lambda\sqrt{n})$$

should contain 95% of the distribution.

Approximation through numerical simulation

We may use simulations in order to validate this approximation.

Assume $\lambda = 1/12{,}000$

```r
X.bar = replicate(10^5,mean(rexp(201,1/12000)))
mean(abs(X.bar-12000) <= 1.96*0.0705*12000)
```

```
## [1] 0.95173
```

Which shows that the Normal approximation is adequate in this example.
How about other values of n or λ?

Approximation through numerical simulation

Simulations may also be used in order to compute probabilities in cases where the Normal approximation does not hold.

Consider the following statistic

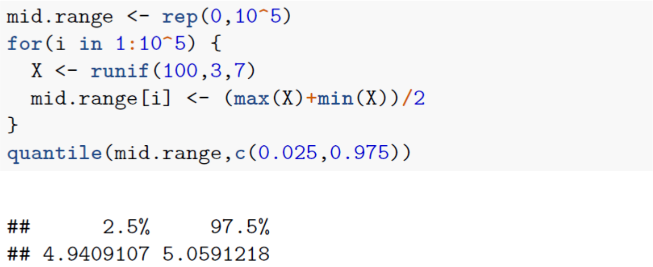
$$(\min(x_i) + \max(x_i))/2$$

where $X_i \sim \mathrm{Uniform}(3,7)$ and $n = 100$

What interval contains 95% of the observations?

Approximation through numerical simulation

Let us carry out the simulation that produces an approximation of the central region that contains 95% of the sampling distribution of the mid-range statistic for the Uniform distribution:



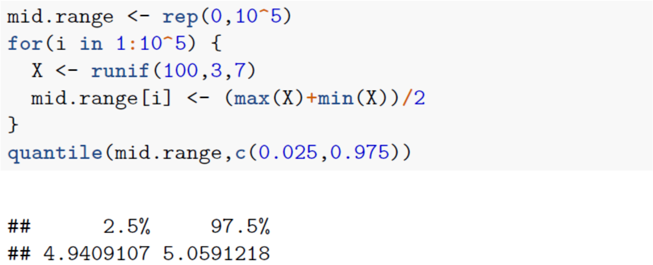
```r
mid.range <- rep(0,10^5)
for(i in 1:10^5) {
  X <- runif(100,3,7)
  mid.range[i] <- (max(X)+min(X))/2
}
quantile(mid.range,c(0.025,0.975))
```

```
##      2.5%     97.5%
## 4.9409107 5.0591218
```

Observe that (approximately) 95% of the sampling distribution of the statistic are in the range [4.941680, 5.059004].

Approximation through numerical simulation

Let us carry out the simulation that produces an approximation of the central region that contains 95% of the sampling distribution of the mid-range statistic for the Uniform distribution:



```r
mid.range <- rep(0,10^5)
for(i in 1:10^5) {
  X <- runif(100,3,7)
  mid.range[i] <- (max(X)+min(X))/2
}
quantile(mid.range,c(0.025,0.975))
```

```
##      2.5%     97.5%
## 4.9409107 5.0591218
```

Observe that (approximately) 95% of the sampling distribution of the statistic are in the range [4.941680, 5.059004].

Approximation through numerical simulation

Simulations can be used in order to compute any numerical summary of the sampling distribution of a statistic.

To obtain the expectation and the standard deviation of the mid-range statistic of a sample of 100 observations from the *Uniform*$(3,7)$ distribution:

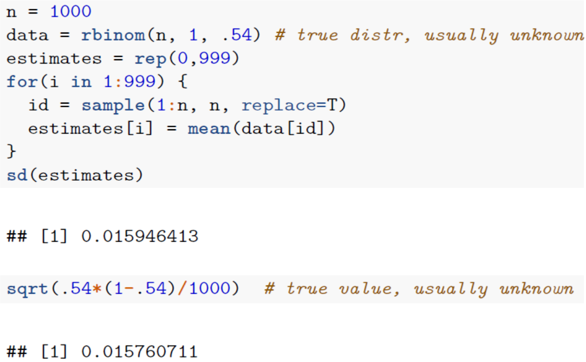
```r
mean(mid.range)
```

```
## [1] 4.9998949
```

```r
sd(mid.range)
```

```
## [1] 0.027876151
```

Approximation through numerical simulation



```r
n = 1000
data = rbinom(n, 1, .54) # true distr, usually unknown
estimates = rep(0,999)
for(i in 1:999) {
  id = sample(1:n, n, replace=T)
  estimates[i] = mean(data[id])
}
sd(estimates)
```

```
## [1] 0.015946413
```

```r
sqrt(.54*(1-.54)/1000)  # true value, usually unknown
```

```
## [1] 0.015760711
```

SIMULATIONS IN PYTHON

Distribution choice flowchart Source

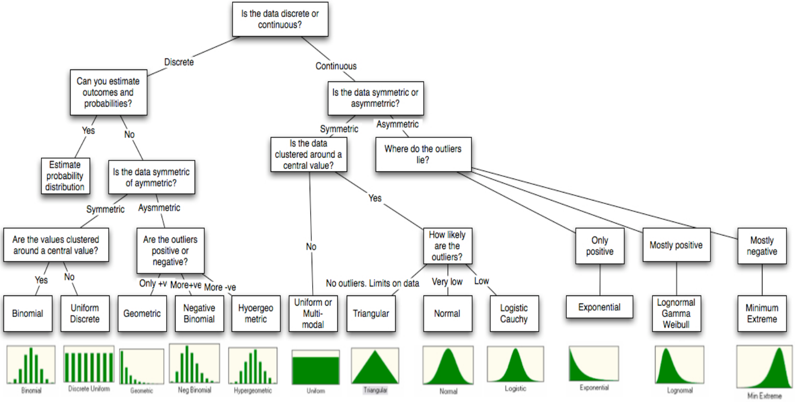


Figure: Source: https://pages.stern.nyu.edu/ adamodar/

Probability Distributions

Application:	Distribution:	Function:
Tossing uneven coin	Bernoulli	scipy.stats.bernoulli
Rolling a dice	Uniform	scipy.stats.randint
Counting errors/successes	Binomial	scipy.stats.binom
Trying until success	Geometric	scipy.stats.geom
Countable, rare events whose occurrence is independent	Poisson	scipy.stats.poisson
Random “noise”, sums of many variables	Normal	scipy.stats.norm

Uniform Distribution

Stochastic Method

```
In [1]:   import scipy.stats

In [2]:   u = scipy.stats.uniform(90, 10)
          u.rvs(5)
Out [2]:    array([93.42, 93.97, . . . , 94.46])

In [3]:   u.rvs(1000).mean()
Out [3]:    94.91863434778571
```

How many values are less than 92?

```
In [4]:   (u.rvs(1000) < 92).sum() / 1000.0
Out [4]:    0.202
```

Analytical Method

```
In [5]:   import sympy.stats

In [6]:   u = sympy.Symbol("u")
          u = sympy.stats.Uniform(u, 90, 100)
          sympy.stats.sample(u)

In [7]:   sympy.stats.E(u)
Out [7]:    95
```

How many values are less than 92?

```
In [8]:   sympy.stats.P(u < 92)
```

Out [8]: $\frac{1}{5}$

Gaussian Distribution

- Consider a Gaussian distribution centered in 5, standard deviation of 1
- Plot the distribution
- Check that half the distribution is located to the left of 5
- Find the first percentile (value of x which has 1% of realizations to the left)
- Check that it is equal to the 99% quantile

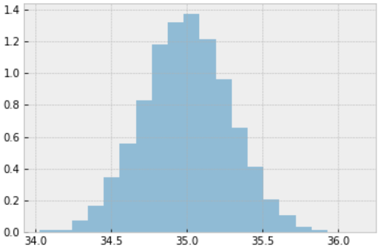
```
In [1]:   import scipy.stats
In [2]:   norm = scipy.stats.norm(5, 1)
           x = numpy.linspace(1, 9, 100)
           plt.plot(x, norm.pdf(x));
In [3]:   norm.cdf(5)
Out [3]:   0.5
In [4]:   norm.ppf(0.5)
Out [4]:   5.0
In [5]:   norm.isf(0.99)
Out [5]:   2.6736521259591592
In [6]:   norm.cdf(norm.isf(0.99))
Out [6]:   0.01
```

Central Limit Theorem

The Central Limit Theorem states that the mean (also true of the sum) of a set of random measurements will tend to a normal distribution, no matter the shape of the original measurement distribution.

```
In [1]:  N = 10000
         sim = numpy.zeros(N)
         for i in range(N):
             sim[i] = numpy.random.uniform(30, 40, 100).mean()
         plt.hist(sim, bins=20, alpha=0.5, density=True);
```

Out [1]:

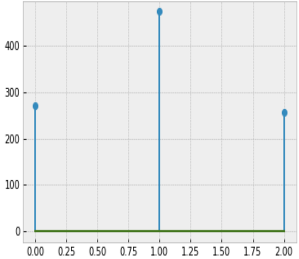


Coin Toss Experiment

Simulate the number of heads when tossing a coin twice. Do this 1000 times, calculate the expected value of the number of heads, and plot the distribution of results (the Probability Mass Function).

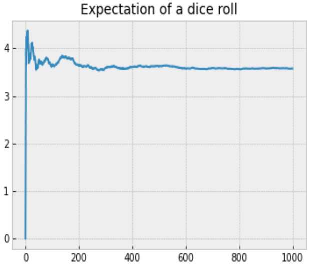
```
In [1]:  N = 1000
         heads = numpy.zeros(N, dtype=int)
         for i in range(N):
            heads[i] = numpy.random.randint(0, 2, 2).sum()
         heads.mean()
Out [1]:   0.987
```

```
In [2]:   values, counts = numpy.unique(heads,
                                        return_counts=True)
plt.stem(values, counts, use_line_collection=True)
plt.show()
```



Rolling a Die

```
In [1]:  N = 1000
         roll = numpy.zeros(N, dtype=int)
         expectation = numpy.zeros(N)
         for i in range(N):
             roll[i] = numpy.random.randint(1, 7)
         for i in range(1, N):
             expectation[i] = numpy.mean(roll[0:i])
         plt.plot(expectation)
         plt.title("Expectation of a dice roll")
         plt.show()
```



Simulating Dice Throws

- Simulate thousand dice throws.
- What is the probability of a die rolling 4?
- What is the probability of rolling 4 or below?
- What is the probability of rolling between 2 and 4 (inclusive)?

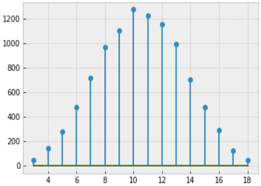
```
In [1]:   dice = scipy.stats.randint(1, 7)
In [2]:   dice.pmf(4)
Out [2]:  0.166
In [3]:   dice.cdf(4)
Out [3]:  0.666
In [4]:   dice.cdf(4) - dice.cdf(1)
Out [4]:  0.5
```

Rolling Three Dice

Estimate the expected value and the probability mass function for the sum of three dice.

```
In [1]:  N = 10000
         roll = numpy.zeros(N, dtype=int)
         for i in range(N):
             roll[i] = scipy.stats.randint(1, 7).rvs(3).sum()
         roll.mean()
Out [1]:  10.5018
```

```
In [2]:  values, counts = numpy.unique(roll, return_counts=True)
         roll = numpy.zeros(N, dtype=int)
         for i in range(N):
             roll[i] = scipy.stats.randint(1, 7).rvs(3).sum()
         roll.mean()
```



The Last Banana Thought Experiment

Two people are stranded on an island with only one banana to eat. To decide who gets it, they agree to play a game. Each of them will roll a fair 6-sided dice. If the largest number rolled is a 1, 2, 3, or 4, then Player 1 gets the banana. If the largest number rolled is a 5 or 6, then Player 2 gets it. Which player has the better chance?

```
In [1]:  N = 1000
         player1 = 0
         for i in range(N):
             roll1 = scipy.stats.randint(1, 7).rvs(1)
             roll2 = scipy.stats.randint(1, 7).rvs(1)
             if max(roll1, roll2) <= 4:
                     player1 += 1
         player1 / float(N)

Out [1]:  0.439
```

Player 1 gets the banana when both the dice rolls are smaller than 5. The probability of a roll smaller than 5 is 4/6, the two dice rolls are independent events, so their combined probability is the product of their individual probabilities: Player 1 gets the banana with probability $\frac{4}{6} \times \frac{4}{6} = 0.444$.

The gambles of the Chevalier de Méré

- Antoine Gombaud, chevalier de Méré, a french gambler and the mathematician Blaise Pascal discussed a problem.
- This discussion was a little daring for the time, because it went against established doctrine of the Catholic church that people should not attempt to predict the future (an activity that was reserved for the deity).
- The discussion and collaboration of Pascal with another mathematician, Pierre de Fermat, helped in the development of probability theory.

The first problem: is it a good idea to gamble on the appearance of at least one 6 when a dice is thrown 4 times?

The second problem: is it a good idea to gamble on the appearance of at least one double six when two dice are thrown 24 times?

The gambles of the Chevalier de Méré

The first problem: is it a good idea to gamble on the appearance of at least one 6 when a dice is thrown 4 times?

The probability of losing this gamble is easy to calculate analytically: each throw has 5 chances out of 6 of not seeing a 6, and the events are independent. So the probability of winning is

```
In [1]:   1 - (5/6.0)**4
Out [1]:   0.517746913580247
```

The probability of winning is greater than 0.5. We can also check this problem with SymPy:

```
In [2]:   import sympy.stats
          D1 = sympy.stats.Die(”D1”, 6)
          D2 = sympy.stats.Die(”D2”, 6)
          D3 = sympy.stats.Die(”D3”, 6)
          D4 = sympy.stats.Die(”D4”, 6)
          sympy.stats.P(sympy.Or(D1 > 5, sympy.Or(D2 > 5, sympy.Or(D3 > 5,
D4 > 5)))) + 0.0
Out [2]:   0.517746913580247
```

The gambles of the Chevalier de Méré

The second problem: is it a good idea to gamble on the appearance of at least one double six when two dice are thrown 24 times?
The probability of losing this gamble is also easy to calculate: there are 35 chances out of 36 (6 * 6) of not seeing a double 6 on each double throw, so the probability of winning is

```
In [1]:  1 - (35/36.0)**24
Out [1]:  0.491403876130903
```

So this is not a good gamble. We can also calculate this analytically with SymPy, by modelling a Binomial random variable:

```
In [2]:  import sympy.stats
         A = sympy.stats.Die("A", 6)
         B = sympy.stats.Die("B", 6)
         doublesix = sympy.stats.Binomial("DoubleSix", 24,
         sympy.stats.P(sympy.And(A > 5, B > 5)))
         sympy.stats.P(doublesix >= 1) + 0.0
Out [2]:  0.491403876130903
```

Pepys and Newton

Samuel Pepys was a great diarist of the English language and a friend of Isaac Newton's. Pepys was a gambler and wrote to Newton to ask which of three events is the most likely:

- at least one six comes up when six fair dice are rolled;
- at least two sixes come up when 12 dice are rolled;
- at least three sixes come up when 18 dice are rolled.

Now Newton wasn't able to use Python, and spent a while working out the answers, but we can stand on the shoulders of giants and figure this out quite easily.

Pepys and Newton

Possibility 1: the probability of rolling at least one six is 1 minus the probability of zero sixes, which is

In [1]: $1-(\frac{5}{6})^6$

Out [1]: 0.665102023319616

Pepys and Newton

Possibility 2: the probability of at least two sixes is 1 minus the probability of zero sixes, minus the probability of a single six. The probability of zero sixes is easy to calculate; here are two ways of calculating it.

Throw a non-6 12 times

In [1]: $(\frac{5}{6})^{12}$
Out [1]: *0.112156654784615*

Probability mass at 0 of a binomial distribution with n=12, p=$\frac{1}{6}$

In [2]: *scipy.stats.binom(12, 1/6.0).pmf(0)*
Out [2]: *0.112156654784615*

So the final answer is
In [3]: *roll12 = scipy.stats.binom(12, 1/6.0)*
1 - roll12.pmf(0) - roll12.pmf(1)
Out [3]: *0.618667373732309*

Pepys and Newton

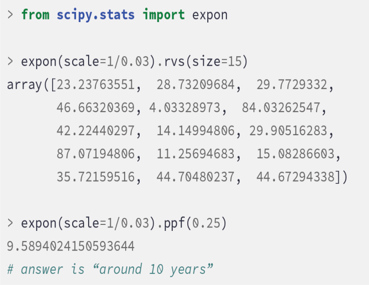
Possibility 3: in the same way, the probability of at least three sixes when rolling 18 dice is one minus the probability of zero sixes, minus the probability of one six, minus the probability of two sixes.

```
In [1]:   roll18 = scipy.stats.binom(18, 1/6.0)
          1 - roll18.pmf(0) - roll18.pmf(1) - roll18.pmf(2)

Out [1]:   0.597345685947723
```

Example: Modeling Earthquake

Simulating earthquake occurrences

- Suppose we live in an area where there are typically 0.03 earthquakes of intensity 5 or more per year.
- Assume earthquake arrival is a Poisson process
  - interval between earthquakes follows an exponential distribution
  - events are independent
- Simulate the random intervals between the next earthquakes of intensity 5 or greater.
- What is the 25-th percentile of the interval between 5+ earthquakes?

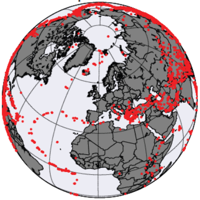


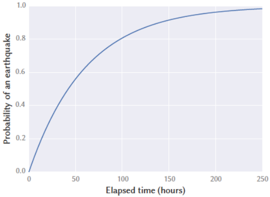
```
> from scipy.stats import expon

> expon(scale=1/0.03).rvs(size=15)
array([23.23763551,  28.73209684,  29.7729332,
       46.66320369, 4.03328973,  84.03262547,
       42.22440297,  14.14994806, 29.90516283,
       87.07194806,  11.25694683,  15.08286603,
       35.72159516,  44.70480237,  44.67294338])

> expon(scale=1/0.03).ppf(0.25)
9.5894024150593644
# answer is “around 10 years”
```

Simulating earthquake occurrences

- Worldwide: 144 earthquakes of magnitude 6 or greater in 2013 (one every 60.8 hours on average)
- Rate: $= 1/60.8$ per hour
- What's the probability that an earthquake of magnitude 6 or greater will occur (worldwide) in the next day?
  - right: plot of the cdf of the corresponding exponential distribution
  - $scipy.stats.expon(scale = 60.8).cdf(24) = 0.326$
- Data source: `earthquake.usgs.gov/earthquakes/search/`

Earthquake locations





Codes and Implementation

Course GitHub: https://github.com/tanujit123/MATH350

Other resources:

- SciPy lecture notes: scipy-lectures.org
- Book "Statistics done wrong" available online at statisticsdonewrong.com
- Risk Analysis course at risk-engineering.org