MATH 157: Mathematics in the world

Homework 7 (Due April 2rd, 2019 at 1:00PM)

Problem 1

We say that a random variable X has the *Poisson distribution* with parameter $\lambda > 0$ if it takes values in the non-negative integers and

$$\mathbb{P}(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

for $k \geq 0$. [1]

1. Show that $\mathbb{E}(X) = \lambda$.
2. Show that $\mathbb{E}(X(X-1)) = \lambda^2$. Use this result to deduce that $\mathbb{E}(X^2) = \lambda^2 + \lambda$ and $Var(X) = \lambda$.
3. What is the probability that X is even? [2]
4. Let X_1 and X_2 be independent Poisson variables with parameters $\lambda_1, \lambda_2 > 0$ respectively. Show that the sum $X = X_1 + X_2$ is Poisson with parameter $\lambda = \lambda_1 + \lambda_2$. [3]
5. Show that if $X_1, \dots, X_n$ are Poisson variables with respective parameters $\lambda_1, \dots, \lambda_n > 0$, then the sum $X = \sum_{i=1}^n X_i$ is Poisson with parameter $\lambda = \sum_{i=1}^n \lambda_i$. [4]

[1] It is not immediately obvious that the total probability is 1, that is, $\sum_{k\geq 0} \mathbb{P}(X = k) = 1$. This follows from the expansion

$$e^x = \sum_{k\geq 0} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots.$$

[2] Hint: Use the expansions of e^{λ} and $e^{-\lambda}$.

[3] Hint: Evaluate the sum

$$\mathbb{P}(X = k) = \sum_{\substack{k_1+k_2=k \\ k_1,k_2\geq 0}} \mathbb{P}(X_1 = k_1, X_2 = k_2) = \sum_{\substack{k_1+k_2=k \\ k_1,k_2\geq 0}} \mathbb{P}(X_1 = k_1) \cdot \mathbb{P}(X_2 = k_2).$$

[4] Hint: Use induction and the previous part.

Problem 2

The variance of a random variable X is given by

$$Var(X) = \mathbb{E}[(X - \mathbb{E}(X))^2].$$

We can generalize this notion by defining the *covariance* of two variables X and Y as

$$Cov(X, Y) = \mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))].$$

The *correlation* of X and Y is a normalized measure of covariance defined as

$$Cor(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}}.$$

Note that both covariance and correlation are symmetric in their inputs:

$$Cov(X, Y) = Cov(Y, X), \qquad Cor(X, Y) = Cor(Y, X).$$

1. Show that $Var(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2$ [5]
2. Show that $\mathbb{E}(X)^2 \leq \mathbb{E}(X^2)$. [6]
3. Show that $Cov(X, Y) = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y)$.
4. Show that $|Cor(X, Y)| \leq 1$. [7]
5. Let X, Y be Random variables, find an expression of $Var(aX + bY)$ in terms of $a, b, Var(X), Var(Y)$ and $Cov(X, Y)$ and prove it.

Problem 3

In class, we talked a little about portfolio theory. In this exercise, we will explore the application of correlation in *modern portfolio theory*. Before getting to the details of the problem we will introduce some basic notions from finance.

The *stock* (shares) of a company are units of fractional ownership. For example, if a company A issues 1 million shares, and an investor purchases $10,000$ of them, they will own 1% of A. While smaller businesses are typically owned by only a few people, larger

[5] Hint: You may use the formula $\mathbb{E}(aX + b) = a\mathbb{E}(X) + b$ for constants a, b.

[6] Hint: Use the two expressions for variance above together with the fact that if $V \geq 0$ is a non-negative random variable, then $\mathbb{E}(V) \geq 0$.

Remark: This is a special case of *Jensen's inequality* which states that if ϕ is a convex function, then $\phi(\mathbb{E}(X)) \leq \mathbb{E}(\phi(X))$ for all random variables X.

[7] Hint: It suffices to show that $Cov(X, Y)^2 \leq Var(X)Var(Y)$. We can further reduce to the case that $\mathbb{E}(X) = \mathbb{E}(Y) = 0$ (adding a constant change the variance or covariance. The statement then reduces to $\mathbb{E}(XY)^2 \leq \mathbb{E}(X^2)\mathbb{E}(Y^2)$ which is the famous *Cauchy-Schwartz inequality*, a fact you can assume.

companies prefer (or are forced to) allow *public ownership*. This means that anyone can purchase the company's stock, typically though an *exchange*.

By fractional ownership, the price of a share of A is related to the value of the entire company; in turn, that is tied to the performance of the underlying business. Put differently, the stock price of A tracks the public's expectation and enthusiasm for the same company. If we believe A is going to perform well, we can express this view by purchasing its stock. If the price goes up, we can later sell it back and profit from the transaction. Ownership of some amount of stock is called a *long (position)* or *going long*.

While more counter-intuitive, there is a way to own a *negative* amount of stock. This is called a *short (position)* or *going short*. Roughly speaking, the mechanics of a short go as follows. Imagine stock A costs \$100 today, and we believe the price will fall to \$90 tomorrow. To profit \$10 from this knowledge, we would like to buy -1 shares today and sell them back tomorrow. First, we find someone who already owns 1 share of A and has no plans to trade it during the next day (e.g., a long-term investor). We enter an agreement with them to borrow this share today. After we receive the share from our lender, we sell it to the market today, and earn \$100. If the price goes down to \$90 tomorrow, we will buy it back and return the share to the lender. As you can see, this transaction earns us \$10. Putting the details of the transaction aside, the bottom line is that abstractly one can own a negative amount of any asset. That said, short positions often increase risk, so it is natural that some investors use long-only strategies.

When investing in larger quantities, the absolute price of a stock is rarely relevant. The important piece of information is the relative change of price over time. The *return* of a stock over a period of time is the relative change in value. For example, if the prices today and tomorrow are respectively P_0 and P_1, the associated return is $r = (P_1 - P_0)/P_0$. When multiple successive periods are studied, using the *logarithmic return* $\ln(P_1/P_0)$ instead makes return figures additive. Note that the regular return is the first order Taylor approximation to the logarithmic return, so for P_1 relatively close to P_0, we expect that the two notions produce virtually identical figures.

The price of any stock in the future is an unknown, so we can model it as a random variable. Likewise, future returns can also be treated as random variables. In this setting, it is important to have measures for the expected return and its standard deviation, also called the *risk*. The *correlation* of two stocks refers to the correlation of the returns of the two stocks.

Our discussion so far was focused on a single stock, but the reality is there are quite a few of them out there (there are around 4000 exchange traded companies in the US). We are naturally led to the following guiding question: If you receive a large sum of money, how would you distribute your investments? Even if we assume that we have solved the extremely difficult problem of computing the expected returns of all stocks are are interested in, their respective risks, and the associated correlation matrix, figuring the amount we should invest in each is not trivial. This problem lies at the center of portfolio theory which won the 1990 Nobel Prize in Economics for Harry Markowitz.

We will use this problem to illustrate the portfolio question for two stocks, A and B.

Assume the expected return of A is 0.1 and of B is 0.2, their risks are respectively 0.1 and 0.2, and they have a correlation of 0.7. Suppose we have a total of \$1 to invest in A and B, and we would like to build portfolios which reflect the following scenarios. [8]

1. Find the return-maximizing long-only portfolio.
2. Find the risk-minimizing portfolio. Find the risk-minimizing long-only portfolio. [9]
3. How do the answers of the previous question change if the correlation is -0.7 instead? [10]
4. Show that a portfolio can achieve arbitrarily large expected returns. What happens to the risk as the returns increase?
5. Find the return-maximizing portfolio whose risk is bounded by 0.3.
6. The previous few parts should have convinced you that risk or returns alone rarely tell the whole story. In practice, a better metric is the *information ratio* defined as the ratio of expected returns to risk. [11]

   Find the portfolio with maximal information ratio.

Problem 4

If we choose n points randomly on a fixed circle, what is the probability they will lie in a semicircle? [12]

Problem 5

In class, we have used a geometric method to compute π. In this problem, we will write a program to implement the method.

Given two independent uniform random variables X and Y, we have

$$\mathbb{P}(X^2 + Y^2 \leq 1) = \frac{\pi}{4}.$$

[8] We are assuming it is possible to own a fractional amount of a share. While this is not true in reality, the problem goes away once investment capital grows.

The portfolio question boils down to finding numbers a and b (the holdings of A and B respectively) such that $a + b = 1$, and they are optimal in a pre-defined sense.

The answers to the following questions are often not nice round numbers, so feel free to use 3 decimal point approximations.

[9] You may find the Question 2 of this PSet helpful

[10] The correlation is different only for this part.

[11] The information ratio is closely related to another metric called the *Sharpe ratio*. While there are some important differences between the two, the two notions agree within the simplified context of the problem.

[12] If you are doing this problem early, we will discuss this problem with $n = 3$ on Thursday.

In class, we explained that this technique can be used to approximate π. The goal of the present problem is to carry out this plan, and investigate the associated error with these approximations.

Before going into the details of the task, we will take the opportunity to discuss some of the advantages vector programming (using NumPy). Consider the following code.

```
import numpy as np
from timeit import timeit

n     = 10**6
times = 10

f1 = lambda: [x**2 for x in xrange(n)]
f2 = lambda: np.arange(n)**2

print(timeit(f1, number = times))
print(timeit(f2, number = times))

## Prints:
## 1.14829111099
## 0.0352470874786
```

The content of the two functions `f1` and `f2` is virtually identical. Each of them computes the squares of the first 10^6 non-negative integers. The two lines involving `timeit` run each of the functions 10 times, and print the time it takes to do so (in seconds). As you can see `f1` is significantly slower than `f2`. In fact, `f1` is more than 30 times slower than `f2`. The point is that using loops (e.g., while, for, list comprehension) is much slower than the built-in functionality of NumPy. Of course, `arange` and the power operator `**` are implemented using loops inside NumPy, but these are highly optimized and take advantage of modern CPU features not available to Python directly. In addition, we can argue that the function `f2` is conceptually simpler. The bottom line is one should always use the available vector primitives when possible.

1. Write a function `pi4(k, n)` which returns a $k \times n$ matrix each of whose entries are 0 or 1, and the latter of these happens with probability $\pi/4$. Don't use any loops. Don't use explicitly the value of π. [13]

2. Write a function `row_cumulative_mean` which given a matrix X returns the row-wise,

[13] Hint: `np.random.rand` returns a matrix of uniformly distributed random numbers in $[0, 1)$ which you can use for both X and Y. You can compare matrices with `np.less`, and change the type of a matrix with `astype`.

cumulative means. For example, if

$$X = \begin{pmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{pmatrix},$$

then the output should be

$$\begin{pmatrix} X_{11} & \frac{X_{11}+X_{12}}{2} & \frac{X_{11}+X_{12}+X_{13}}{3} \\ X_{21} & \frac{X_{21}+X_{22}}{2} & \frac{X_{21}+X_{22}+X_{23}}{3} \\ X_{31} & \frac{X_{31}+X_{32}}{2} & \frac{X_{31}+X_{32}+X_{33}}{3} \end{pmatrix}.$$

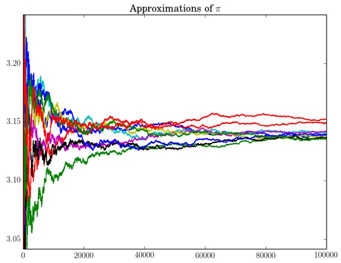
Don't use any loops. [14]

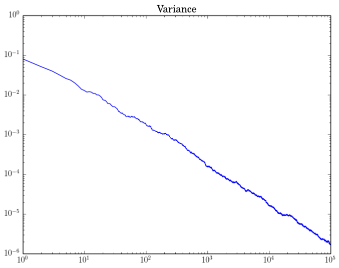
3. We can treat the output of `pi4` as $k \times n$ tests, organized in k experiments (the rows). Computing the row-wise cumulative means, we get a glimpse of how our approximations of $\pi/4$ evolve as we run more and more tests.

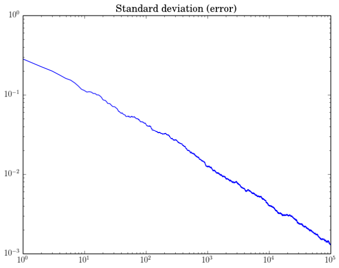
   Use $n = 10^5$ and $k = 10$, and plot the evolution of the approximations of π you get from each of the experiments. Limit the y-axis interval to $(\pi - 0.1, \pi + 0.1)$. [15] You can use a loop to draw the 10 curves here. What are the final approximations you get from each of the 10 experiments?

4. Now that we have approximated π, we would like to investigate how the error evolves as we use more and more tests. Use $n = 10^5$ and $k = 100$ to arrive at a matrix of approximations for $\pi/4$. Plot the column-wise variance on a log-log scale. [16]

   The square root of the variance, also called *standard deviation*, is a good measure of the error in an approximation. Produce the same plot as above with standard deviations. Don't use a loop to compute square roots.







[14] The row-wise cumulative sums can be computed using `np.cumsum`. The result should then be divided by a matrix of the form

$$\begin{pmatrix} 1 & 2 & 3 & \cdots \\ 1 & 2 & 3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

You can get this using `np.fromfunction(lambda x, y: (1 + y), (k, n))`.

[15] See `plt.ylim`.

[16] See `np.var` and `plt.loglog`.

More generally, consider independent identically distributed random variables $T_1, \ldots, T_n$ such that T_i is 1 with probability $p \in (0,1)$ and 0 with probability $1 - p$. If $S_n = 1/n \sum_{i=1}^{n} T_i$, then

$$\begin{aligned} Var(S_n) &= \frac{1}{n^2} Var(\sum_{i=1}^{n} T_i) \\ &= \frac{1}{n^2} \sum_{i=1}^{n} Var(T_i) \\ &= \frac{p(1-p)}{n}. \end{aligned}$$

The important observation is that the variance of S_n, the n-th approximation of p, is $O(1/n)$. This is evident from the log-log plot of the variance you produced. It is close to a line with slope -1. Similarly, the standard deviation is also close to a line but the slope is $-1/2$.

The bottom line is that in its simplest form the Monte Carlo method is very slow. If we want to compute one more decimal place for our approximation, that is, reduce the error by a factor of 10, the necessary computations increase by a factor of 100. The basic form of Monte Carlo we presented is almost never used in practice. There are an array of techniques which improve on this simple idea.