THE LONDON SCHOOL OF ECONOMICS AND POLITICAL SCIENCE

Assignments
ME200 Computational Methods in Financial Mathematics
Summer School 2023

Johannes Ruf and Luitgard A. M. Veraart
Emails: J.Ruf@lse.ac.uk and L.Veraart@lse.ac.uk

This version: 09/06/2023

Assignment 1

Exercise 1. (**) If we roll three fair six-sided dice, which is more likely: a sum of 5 or a sum of 17?

Exercise 2. A round-robin tournament is being held with n tennis players; this means that every player will play against every other player exactly once.

1. How many games are played in total?
2. How many possible outcomes are there for the tournament (the outcome lists of who won and who lost for each game)?

Exercise 3. For five days, you're both playing poker and roulette. These are your winnings/losses in poker for the five days: -30, 40, 10, -10, 40. These are your winnings/losses in roulette for the five days: 10, 20, -60, -10, 0.

1. Store these results in two vectors in Python.
2. What are your overall winnings in poker? What are your overall winnings in roulette? Use Python to compute.
3. For each of the five days, what are your winnings/losses? Use Python to compute.

Exercise 4. In Python, sample 8 numbers between 5 and 9, with replacement, and with probabilities given by (0.1, 0.2, 0.1, 0.05, 0.55).

Hint: To solve this problem, you might have to type `?np.random.choice` into an empty cell and read the documentation for the choice function.

Exercise 5. In Python, check for different values of k and n that the following identities hold.

1. $$\binom{n}{k} = \binom{n}{n-k};$$
2. $$n\binom{n-1}{k-1} = k\binom{n}{k}.$$

Hint: In Python to check whether two quantities agree, we can use two consecutive equalities. E.g., `5 == 4` returns `False` while `8==8` returns `True`.

Assignment 2

Exercise 6. Define the mathematical concept of a random variable. What is a discrete random variable and what is a continuous random variable? (Try to do this without checking the lecture notes.)

Exercise 7. (**) If X, Y, Z are r.v.s such that X and Y are independent and Y and Z are independent, does it follow that X and Z are independent?

Exercise 8.

1. Give an example of dependent r.v.s X and Y such that $P[X < Y] = 1$.
2. Give an example of independent r.v.s X and Y such that $P[X < Y] = 1$.

Exercise 9. Let X be an r.v. with CDF F, and $Y = \mu + \sigma X$, where μ and σ are real numbers with $\sigma > 0$. Find the CDF of Y, in terms of F.

Exercise 10. Generate 100, 1000, and 100000 samples of a uniformly distributed random variable and plot each time a histogram of these samples.

Exercise 11. An exponentially distributed random variable with parameter $\mu > 0$ has the PDF

$$f(x) = \begin{cases} \mu \mathrm{e}^{-\mu x}, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}.$$

Plot this PDF in Python for different values of μ. (Hint: You might google the corresponding Python command for the PDF of this distribution.)

Exercise 12. A doubly exponentially distributed random variable with parameter $\mu > 0$ has the PDF

$$f(x) = \frac{1}{2}\mu \mathrm{e}^{-\mu|x|}.$$

To generate a sample of a doubly exponentially distributed random variable, we may generate a sample X of an exponentially distributed random variable and a sample U of a $\mathrm{DUnif}(\{-1, 1\})$–distributed random variable. Then the product UX is doubly exponentially distributed.

Write Python code to generate a histogram of the doubly exponentially distribution.

Assignment 3

Exercise 13. Define variance and standard deviation of a random variable. (Try to do this without checking the lecture notes.)

Exercise 14. Prove the following two statements for any r.v. X and any constant $c \in \mathbb{R}$:

1. $\mathrm{Var}(X + c) = \mathrm{Var}(X)$.
2. $\mathrm{Var}(cX) = c^2 \mathrm{Var}(X)$.

Exercise 15. (**) Compute the expectation of an exponentially distributed r.v. with parameter μ (recall Exercise 11 for the density).

Exercise 16. In Python, generate 100 samples from the $\mathcal{N}(4, 9)$ distribution and compute the sample mean and sample standard deviation.

Exercise 17. We call a twice differentiable function g convex if $g''(x) \geq 0$ for all real numbers x. The following statement holds: For each convex function g and any random variable (such that the corresponding expectations exist) we have the inequality

$$g(E[X]) \leq E[g(X)].$$

(This inequality is called Jensen's inequality.)

Check in Python that this inequality holds. To do this, consider the examples $g(x) = x^2$ and $g(x) = x^4 + 2x^2$ and samples from the normal and uniform distribution.

Assignment 4

Exercise 18. Try to answer this question without putting the code into a Jupyter notebook.

1. When running the following Python code what will be printed out?

```
import numpy as np

def M(t=0):
    return np.cos(t**2 * np.pi)

print(M())
print(M(1))
```

2. When running the following Python code what will be printed out?

```
import numpy as np

def M(t=0):
    return np.cos(t**2 * np.pi)
    print(t**2)

print(M())
```

Exercise 19. In the lecture, we have used the inverse transform method to generate samples from the exponential distribution with parameter $\mu = 1$. Adapt this code to generate samples from the exponential distribution with arbitrary parameter $\mu > 0$. Moreover, also generate such samples via the NumPy command. Then plot two histograms of the two sets of samples.

Exercise 20. Use the inverse transform method in Python to generate 10000 samples from a discrete random variable X taking values in $\{1, 2, 3, 4\}$ with probability $P[X = i] = ci$, for all $i \in \{1, 2, 3, 4\}$, for some constant $c > 0$ (which needs to be determined). Plot a histogram of these samples.

Hint: You might have to use Python's if-else statement. Search for it only to see how it is working.

Exercise 21. The logistic distribution is characterized by the PDF

$$f(x) = \frac{e^{-x}}{(1 + e^{-x})^2}.$$

Using the inverse transform method, work out the function h, such that $h(U)$ follows the logistic distribution if $U \sim \mathrm{Unif}(0, 1)$. Then, use Python to generate 10000 samples from the logistic distribution and plot them in a histogram.

Exercise 22. Check on `https://en.wikipedia.org/wiki/Fibonacci_number`, what the Fibonacci numbers are. Then write a function in Python that takes as input an integer n and returns the first n Fibonacci numbers.

Hint: You might need a for loop. Also pay special attention to the cases $n = 1$ and $n = 2$.

Assignment 5

Exercise 23. Describe the acceptance-rejection algorithm including its assumptions. (Try to do this without checking the lecture notes.)

Exercise 24. Assume you can generate a standard normally distributed sample X. From X you want to generate a sample $Y \sim \mathcal{N}(\mu, \sigma^2)$ for some real number μ and some $\sigma > 0$. How can you do this?

Exercise 25. In Python, implement the acceptance-rejection algorithm to generate a standard normally distributed sample from the doubly exponential distribution. To obtain the doubly exponential sample, recall the solution of Exercise 12. Generate 10000 samples and plot them in a histogram.

Exercise 26. Recall Exercise 22. Now, write a new function that takes an input an integer k and returns all Fibonacci numbers that are smaller than or equal to k.

Exercise 27. Implement the Box-Muller method to generate 10000 samples from the standard normal distribution and plot a histogram of those samples.

Hint: Using numpy.concatenate can be useful to combine two numpy arrays.

Assignment 6

Exercise 28. In Python, generate 500 independent rolls $X_1, X_2, \cdots, X_{500}$ of a fair die. For each $k \in \{1, 2, \cdots, 500\}$ compute the average $\overline{X}_k = \sum_{j=1}^{k} X_k/k$ and plot $\overline{X}_k$ against k. What do you observe?

Exercise 29. Without checking the lecture notes, compute the variance of the sample mean

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

provided that $X_1, X_2, \cdots$ are i.i.d. with finite variance. What does 'i.i.d.' stand for?

Exercise 30. Consider the function.

$$h(x) = \cos(\mathrm{e}^{3x}).$$

In Python, compute the integral

$$\int_0^1 h(x)dx$$

via Monte-Carlo. Similarly as in Exercise 28 check convergence.

Assignment 7

Exercise 31 (Checking that the risk-neutral probability is a probability)**.** Consider a one-period binomial model and suppose the no-arbitrage condition $d < 1 + r < u$ holds and suppose that $0 < d < u$, define

$$\tilde{p} = \frac{1 + r - d}{u - d}.$$

Show that $\tilde{p} \in (0, 1)$ and $1 - \tilde{p} = \frac{u-1-r}{u-d}$.

Exercise 32 (Sufficient condition for no-arbitrage)**.** Consider a one-period binomial model with $0 < d < u$ and $r > -1$ and assume that

$$d < 1 + r < u$$

holds. Show that there is no arbitrage in the model.

Exercise 33 (Pricing a European put option)**.**

1. Consider a one-period binomial model with model parameters $S_0 = 4$, $u = 2$, $d = \frac{1}{2}$, $r = \frac{1}{4}$. Compute the time-0 price of a European put option with maturity $T = 1$ and strike price $K = 5$ in this model (recall that its payoff at time 1 is $(K - S_1)^+$).

2. Write a function in Python that computes the price of a European put option in the one-period binomial model. (Test that it works by using the analytical price derived in 1.)

Assignment 8

Exercise 34 (Hedging a long position)**.** Consider the one-period binomial model with $S_0 = 4$, $u = 2$, $d = 1/2$, $r = 1/4$ and a European call option with strike price $K = 5$ and maturity $T = 1$ in this model. We have seen in the lecture that this call option has time-0 price $V_0 = 1.2$.

Consider an investor that has a long-position in this European call option, i.e., at time 0 the investor owns the option for which she has paid $V_0 = 1.2$. She wants to earn the interest rate of 25% on this capital until time one (without investing any more money and regardless of the outcome of the coin toss the investor wants to have $\frac{5}{4} \cdot 1.2$ at time 1 after collecting the payoff from the option (if any) at time 1).

How should the investor trade in the stock and the riskless asset to accomplish this?

Exercise 35 (Monte Carlo approximation of European put option)**.**

1. Write a function in Python that generates a sample from the stock price at time 1 in the one-period model using the risk-neutral probabilities. (Hint: Use the method *choice* for this which is available in *numpy.random*.)
2. Use this function to compute a Monte Carlo estimator of the time-0 price of the European put in the one-period binomial model with parameters specified in Exercise 33.
3. Compare the Monte Carlo price to the analytical price.

Exercise 36 (Multiperiod binomial model)**.** Implement the pricing formula (8.17) for a derivative security in the multi-period binomial model in Python for a European call option and for a European put option.

Assignment 9

Exercise 37 (Pricing a given European option)**.** Consider the 2-period binomial model with $S_0 = B_0 = 1$, $u = 3$ and $d = 1/3$.

1. For which interest rates $r \in (-1, \infty)$ is the market free of arbitrage?
2. Now assume that $r = 2/3$. Consider a European option with maturity $T = 2$ and payoff at time 2 given by

$$V_T(\omega) = \begin{cases} 1 - x, & \text{if } \omega = HH, \\ x, & \text{if } \omega = HT, \\ x, & \text{if } \omega = TH, \\ x - \frac{1}{2}, & \text{if } \omega = TT, \end{cases}$$

Thm. 104.4
Section 8.2

where $x \in [\frac{1}{2}, 1]$. Compute the time-0 price of this option.

Exercise 38 (Asian option)**.** Consider a three period binomial model with $S_0 = 4$, $u = 2$, $d = 1/2$, $r = 1/4$. Then, $\tilde{p} = 1/2$. For $n = 0, 1, 2, 3$ define $Y_n = \sum_{k=0}^{n} S_k$. Consider the following *Asian call option* that expires at time $T = 3$ and has strike $K = 4$ and payoff at time 3 given by

$$\left(\frac{1}{4} Y_3 - 4\right)^+ .$$

"Key idea" in page 91
"Tools" in (8.13) and (8.14)
Replica of section 9.1

(This is like a European call option with the exception that the option is based on the average stock price rather than on the terminal stock price.)

Let $v_n(s, y)$ denote the price of this option at time n if $S_n = s$ and $Y_n = y$. (Note in particular, that $v_3(s, y) = \left(\frac{1}{4} y - 4\right)^+$.)

Instead of M_n you need to use Y_n

1. Develop an algorithm for computing v_n recursively. In particular, write a formula for v_n in terms of v_{n+1}. Obtain the appropriate version of (9.1)
2. Apply the algorithm developed in 1. to compute $v_0(4, 4)$. Second half of page 92
3. Provide a formula for $\Delta_n(s, y)$, the number of shares of stocks that should be held by the replicating portfolio at time n if $S_n = s$ and $Y_n = y$. Second half of page 93

Exercise 39 (Generating samples from the Normal distribution)**.** An antiquated generator for the normal distribution is: Generate $U_1, \ldots, U_{12} \sim \text{Unif}\left[-\frac{1}{2}, \frac{1}{2}\right]$. Set $Z = \sum_{i=1}^{12} U_i$.

1. Show that $\mathrm{E}[Z] = 0$ and $\mathrm{Var}(Z) = 1$. Part I
2. Provide an argument why Z could be considered to be an (approximate) sample from the standard normal distribution. CLT
3. Implement the above generator in Python. Use histograms to compare the random numbers generated by the generator above to those generated using an alternative function available in Python for generating normally distributed random variables.

Note: This exercise does not suggest that you should be using the generator above to sample from the standard normal distribution!

Assignment 10

Exercise 40 (Central Limit Theorem)**.** Without looking at the lecture notes state the Central Limit Theorem.

If in trouble: study the statement in the notes, then do exercise 41, then come here again and do this exercise.

Exercise 41 (Illustration of the Central Limit Theorem)**.** Generate samples from the Beta distribution in Python using numpy.random.beta (read the numpy help page for this). Read the Wikipedia page for the Beta distribution to see what it looks like and what its expectation and variance are.

Investigate (using Python) what the empirical CDF of the standardised mean of the samples of the Beta distribution looks like (similarly to the examples discussed in the lecture) for different numbers of samples. From the Central Limit Theorem we know that for large sample size this should look like the CDF of the standard normal distribution.

This should be ok

In addition to the empirical CDF plot histograms of the standardised mean in Python as well.

Exercise 42 (Black-Scholes option pricing formula - European call option)**.** Consider the Black-Scholes market. Compute analytically the time-0 price of a European call option with strike price K, i.e., show that

$$E\left[e^{-rT}(S_T - K)^+\right] = S_0\Phi(D_1) - Ke^{-rT}\Phi(D_1 - \sigma\sqrt{T}),$$

where

$$D_1 = \frac{\log\left(\frac{S_0}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}.$$

Check proof of (10.2) and follow accordingly.
If in trouble (or if completed), do an empirical version of this.
I.e. sample the outcome of the call option and check that on average it has the same value of the close formula.

Assignment 11

Exercise 43 (Black-Scholes option pricing formula - European put option)**.** Consider the Black-Scholes market. Compute analytically the time-0 price of a European put option with strike price K.

Exercise 44 (Variance reduction techniques for option pricing)**.**

1. Write down a Monte Carlo estimator for the time-0 price of a European put option in the Black-Scholes market.
2. Write down an antithetic variates estimator for the time-0 price of a European put option in the Black Scholes model.
3. Write down a control variate estimator for the time-0 price of a European put option in the Black-Scholes model.
4. Write Python code that computes the analytical time 0-price of a European put, the corresponding Monte Carlo, antithetic variates and control variates estimators. Investigate using Python how the performance of the different estimators depends on the strike price.

Assignment 12

Exercise 45 (Result used for computation of the Vega)**.** Let

$$D_1 = \frac{\log\left(\frac{S_t}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}}, \qquad D_2 = D_1 - \sigma\sqrt{T-t}.$$

Show that

study log(f(D_1) / f(D_2))

$$S_t\varphi(D_1) = Ke^{-r(T-t)}\varphi(D_2),$$

where $\varphi(x) = \frac{1}{\sqrt{2\pi}}\exp(-\frac{x^2}{2})$.

Exercise 46 (Bisection method for computing implied volatilities)**.** The Bisection method is another numerical method to approximate the root of a continuous function f, i.e., to find an $x \in \mathbb{R}$ such that $f(x) = 0$.

Just before Thm 155

The Bisection method starts by choosing an interval $[a, b]$, where $a, b \in \mathbb{R}$ with $f(a)f(b) < 0$. Since f is continuous, there has to be at least one $x \in [a, b]$ such that $f(x) = 0$.

Next, a sequence of intervals is constructed (with left boundary l_n and right boundary r_n) by halfing the current interval which gives two new intervals and the interval for which the product of the function values evaluated at the endpoints of the interval remains negative is selected as the next interval. The iteration stops if the interval length is small enough.

The Bisection method can be characterised by the following pseudo code:

```
1. Find $a, b \in \mathbb{R}$, $a < b$, such that $f(a)f(b) < 0$. Set $n = 0$ and $l_n = a$, $r_n = b$.
2. While $r_n - l_n \geq \epsilon$ (where epsilon is a small positive constant that you can choose), do the following:
3. Set $y = \frac{l_n + r_n}{2}$.
4. If $f(l_n)f(y) < 0$ set $l_{n+1} = l_n$ and $r_{n+1} = y$. If $f(y)f(r_n) < 0$ set $l_n = y$ and $r_{n+1} = r_n$.
5. Increase $n$ to $n + 1$ and go to 2.)
6. Return $y$ as the approximation of the root of $f$.
```

Write Python code that computes implied volatilities corresponding to the European Call option in the Black-Scholes model using the Bisection method. Use the example from the lecture (where we used Newton's method) to check your code.

State one advantage and one disadvantage of the Bisection method compared to the Newton method for finding the root of a function.