Assignments

Nr.	Topic	Lecturer	Assigment	Points
1.	Option Basics, BSM model	Erzsébet Romsics	a.) Smart up our plotter to be able to customize the hard-coded layout setup inside the function. Add more custom features that you would like to use. b.) Upgrade the BSM call option pricer to be able to return the greeks as well, not just the price. c.) Implement the BSM pricer for European put option. d.) With the call and put pricer, check if Put-Call parity holds in practice.	4 3 3 5
2.	Building a library	Gábor Friedmann	a.) Get familiar with digital options and implement EuropeanDigitalContract and EuropeanDigitalAnalyticPricer (fair value and greeks) in the library (src folder). Implement unit test for fair value using pytest (test folder). b.) Get familiar with bull spread options and by utilizing EuropeanDigitalAnalyticPricer and EuropeanAnalyticPricer, create a figure to visualize that a digital call option can be thought of as a limit of bull spread call options.	12 8
3.	FlatVol	Gábor Friedmann	Introduce two class variables in Pricer that store the finite difference method (forward, backward, central) to be used in numerical greek calculation, and whether the bump size is absolute or relative. Create two Enums to store the labels. Implement all combinations for delta and gamma calculation. Compare greeks using the analytic and the finite difference methods on a set of European options and visualize the numerical error. Compare absolute and relative bumping and suggest optimal bump sizes. Work with the code in src folder.	15
4.	Finite methods intro	Ádám Farkas	We have $S_1(t), S_2(t), \ldots, S_n(t)$ independent stock processes with equal volatility, $S_i(0) = 100$ for all $i \in \{1, \ldots, n\}$. The contract at maturity pays the value of the most expensive stock: $V(T) = \max_{1 \le i \le n} S_i(T)$. a.) What is the formula and numerical solution for $V(0)$? b.) Implement this as a finite method. How many stock processes are computationally feasible?	20
5.	Tree methods	Lajos Vágó	Implement calibrator for Tree. Background: In the Black–Scholes model volatility is a key parameter that determines option price. In a BalancedBinomialTree the same role is played by step size: higher step is like higher volatility, it results in higher option prices. Also, there is a 1-1 correspondence between step size and price if all other parameters are fixed. Task: Write a calibrate method which, for a given "nr_steps" parameter, computes the "vol" for which a BalancedBinomialTree model returns the same price as the Black-Scholes analytic formula does for a given European call. (See the __init__ method of TreeParams in numerical_method.py). The method should update the TreeParams of the TreePricer. Hint: To do it you will need to use a root solver, I recommend using scipy.optimize.minimize.	15

Nr.	Topic	Lecturer	Assigment	Points
6.	Tree methods	Lajos Vágó	Implement a TreePricer for up-and-out call Barrier option with a continuous barrier and make a few comparisons to the analytic implementation, see details below. Background: Up-and-out call Barrier options are introduced in one of the Monte Carlo clases. Task: Create a pricer so that we can price these contracts with the Tree method as well. Then compare prices to MC for given parameters : Strike = spot, exp = 1Y, nr steps = 252, and a few Barrier levels: 1.1, 1.2, 1.3. Hint: To create the pricer you just have to create a new child of TreePricer with the appropriate pre_final_value method. For "vol" parameter of TreeParams just use the same vol you use in the analytic formula.	20
6.	Monte Carlo methods	Ernő Solymosi	Implement and test the Milstein scheme a.) Implement the Milstein scheme in the evolve_simulated_spot method, validate the implementation by pricing an option with it and compare the result with analytic. b.) Compare the terminal distribution of a stock using Euler and Milstein method against the theoretical distribution (lognormal). Demonstrate the convergence of both methods by increasing the number of steps.	7 8
7.	Monte Carlo methods	Ernő Solymosi	Improve delta calculation for FlatVol MonteCarlo In BS model the stock price at time t is $S_t = S_0 e^{(r-\sigma^2/2)t+\sigma dW_t}$. Knowing this, when we bump the initial spot price S_0 for delta calculation, we do not have to resimulate the spot paths, we can scale the already simulated spots. a.) Implement the improved delta calculation for GenericMCPricer delta for FlatVol market model. b.) compare the result and calculation time of the improved delta with the default bump and reval delta.	10 5
8.	Asian options	László Varga	a.) Show by simulation and a graph that for increasing volatility, the effectiveness of the moment matching approximation deteriorates. b.) What would you expect about the effectiveness of the moment matching method, if the length of the averaging period tends to infinity? Create a test to assess it. (Hint: think on a famous theorem in probability theory.) c.) Devise and implement a proper unit test for the moment matching Asian option pricer.	6 7 7
9.	Barrier options	László Varga	Implement and test the upper barrier put option pricers under continuous monitoring. a.) Implement the barrier up-and-in put option (UIP) and up-and-out put option (UOP) pricers in the codebase. b.) Where do the prices of UIP/UOP options converge, if the barrier tends to infinity? Create a graph to visualize the results. c.) Create a unit test that asserts the in-out parity for UIP and UOP barrier options.	7 4 4