Introduction to Quantitative Finance and Financial Risk

2nd Assignment: Pricing an American Straddle

Tsirmpas Dimitris f3352315

Notes

- There are two scripts; option_pricing.*py* which is *executable* and option_lib.*py,* which contains the pricing model implementations.
- The project requires no dependencies.
- Additional documentation and comments can be found in the source code.
- All algorithms follow a simple class hierarchy by implementing the abstract `Option` class defined in option_lib.py and using the `OptionType` Enum in the same file.
- Since the option is at-the-money, $K = F_t = S_0V(0, 1y)$

American option pricing

We price American Options via a binomial pricing model defined in the class `AmericanOption`. The class implements the abstract method `Option.price()` by constructing two trees; one holding the prices of the stock, and the other the prices of the option for each time-step.

Trees are implemented using Python lists. Each tree is a list and each level of the tree is a new, nested list. Since the tree is binomial (each node has always two branches), we connect the nodes implicitly by index; node 2 of level 1 will be always connected with nodes 2*2 and 2*(2+1) of level 2 (next nested list). Note that this is technically redundant, since binomial pricing models are *recombining* trees. We choose the above method for the sake of programming simplicity.

The price computation begins by constructing the stock price tree (forward pass). Since the compounding term ($\mathrm{e}^{(r-\delta)h}$) is also used elsewhere, we split the stock growth term $e^{(r-\delta)*h \pm \sigma*\sqrt{(h)}}$ into the compounding and uncertainty term, which we multiply / divide accordingly to get the `u` and `d` factors. This is computationally inefficient (since we evaluate the exponential function twice) but much easier to work with programmatically. This procedure can be found in the `_compound()` and `_uncertainty()` methods. Having defined these methods, we work forwards until the tree is completed.

The last step of the price computation is the backwards pass through the stock price tree. For each node in the stock price tree, we compute the payoff for that option, defined as max(payoff_hold, payoff_exercise). Payoff_exercise is defined in the `Option._payoff()` superclass method, and

payoff_hold in `AmericanOption._hold_payoff()`. In reality, we construct a second tree, treating the stock price tree as read-only, which contains the prices of the option for each corresponding node. The root of that tree is returned as the option's current price.

Both forward and backward trees are cached in the object. Thus, their values are calculated once and when needed again, instantly returned. This significantly reduces computational times, especially for the option Vega and Delta computations.

Computing the option Delta is trivial, since we already hold the forward and backward trees. Calculating Vega is done by creating a new `AmericanOption` with σ' = σ * 1.01, calculating its price and returning (price' – price) / 1.01. These can be found in the respective `AmericanOption.delta()` and `AmericanOption.vega()` methods.

European option pricing

We employ a Monte Carlo pricing model for European Options. The implementation can be found in the `EuropeanOption` class.

We implement the abstract `Option.price()` method by iteratively sampling a standardized normal distribution. The method `EuropeanOption._calculate_rand_price()` accepts the normal sample and computes the random price path. The result is discounted to the present, and added to a running sum in `EuropeanOption.price()`. We extract the expected option price by averaging the discounted price sum.

Note that we could have used the `numpy` library to significantly speed up computations by exploiting the vectorized computations and continuous storage provided by the library. We chose to use only functions from the python standard library to keep the project clear of dependencies.

Output

American Call Price: 3.8451459731566704, Delta: 0.6038651166625677, Vega: 3.6161203303377576

American Put Price: 3.6459991371073213, Delta: -0.40590474947503713, Vega: 3.625580527607042

European Call Price: 1.73329809677804

European Call Price: 2.19752811114358

Check: European options must be cheaper than American: True

Check: Call Delta - Put Delta ~= 1: Result= 1.0097698661376049

Process finished with exit code 0