1   MC_lookback.py — Floating-Strike Lookbacks

Purpose

Simulates several asset price paths and tracks the running maximum and minimum to show the payoff for **floating-strike lookback calls and puts**.

Learning focus

- **Lookback call payoff** $= max(0, final_{price} - minimum_{price_{seen}})$

- **Lookback put payoff** $= max(0, maximum_{price_{seen}} - final_{price})$

- The "strike" is not fixed. It is determined by the most advantageous observed price over the life of the option.

- Both calls and puts gain from being able to choose the best price retrospectively.

Key intuition

Because the holder can lock in the most favourable strike in hindsight, floating-strike lookbacks are more valuable than standard options. Their payoff is entirely **path-dependent** — the history of prices matters as much as the final level.

Experiment

Increase sigma (volatility) to see the gap between running max and running min grow, which increases potential lookback payoffs.

2   Payoff_vanilla_lookback.py — Fixed-Strike Lookbacks

Purpose

Shows how a **fixed-strike lookback call** compares to a standard vanilla call for one simulated path.

Learning focus

- **Vanilla call payoff** $= max(S_T - K, 0)$

- **Fixed-strike lookback call payoff** $= max(S_T - min_{price_{seen}}, 0)$

- The strike is fixed at inception, but the lookback call benefits from knowing the lowest price over the life of the option, effectively giving it a built-in advantage compared to the vanilla call.

Key intuition

In a fixed-strike lookback, the option holder still has a set strike but benefits if the underlying traded much lower during the life of the option — this increases the likelihood of being in the money and enlarges the payoff when it is.

Experiment

Run multiple paths and choose cases where the final price is below strike but well above the minimum price seen. You will notice that the vanilla call payoff is zero in such cases, but the lookback call still delivers a positive payout.