Reflections on Option Pricing Models

Tom Beaugé

January 27, 2025

Abstract

This document reflects on my self-directed exploration of financial models for option pricing, beginning with the single-period binomial tree. I discuss the theoretical foundations, implementation challenges, and insights gained from constructing and analyzing the model. This work serves as a foundation for future investigations into more complex pricing frameworks, such as multi-period trees, stochastic calculus, and market microstructure.

1 Introduction

The pricing of financial derivatives represents a cornerstone of modern quantitative finance. My project begins with the simplest discrete-time model—the binomial tree—to build intuition about no-arbitrage pricing, risk-neutral valuation, and dynamic hedging. Though elementary, this model encodes key principles that generalize to continuous-time frameworks like the Black-Scholes-Merton model. My goal is to iteratively implement and analyze increasingly sophisticated models, using each step to deepen my understanding of stochastic processes, measure theory, and computational finance.

2 The Binomial Tree Model

With its very straightforward concept and basic mathematics, the binomial model seemed like a logical introduction to options pricing. However, this overtly simplistic binary idea that the price of an option can either go up or down is something to be noted. In the words of Paul Wilmott: "the model is for demonstration purposes only, it is not the real thing. As a model of the financial world it is too simplistic, as a concept for pricing it lacks the elegance that makes other methods preferable, and as a numerical scheme it is prehistoric. Use once and then throw away." This word of caution was kept in mind while I sought to implement it based on Chapter 15 of his book *Paul Wilmott Introduces Quantitative Finance*.

2.1 A Simple, One-step Binomial Tree

In essence, the binomial model, introduced by Cox, Ross, and Rubinstein (1979), provides a discrete-time approximation of asset price dynamics, such that the underlying asset price S_t evolves in a probability space as:

$$S_{t+1} = \begin{cases} S_t \cdot u & \text{with probability } p, \\ S_t \cdot d & \text{with probability } 1-p. \end{cases}$$

where $u > 1 + r > d > 0$ ensures arbitrage-free dynamics, and r is the risk-free rate. Its simplicity belies its ability to replicate derivative payoffs through a self-financing portfolio of the underlying asset and risk-free bonds.

2.1.1 Implementation

From what was understood, the main idea of the binomial model is to create a risk-neutral valuation. This was achieved by constructing a portfolio composed of the underlying asset and risk-free bonds to exactly match the option's payoff. To prevent any arbitrage, the option's price must thus correspond to the cost of replicating this portfolio. The methodology for constructing this risk-free portfolio was directly inspired by this YouTube video, with the slight addition of fixed interest rates to account for the time value of money.

The relationship governing the replicating portfolio is expressed as:

$$P_t = \Delta S_t - V_t$$

Here, P_t represents the value of the portfolio at time t, Δ is the hedge ratio indicating the number of units of the underlying asset held, S_t is the current price of the underlying asset, and V_t is the current price of the option. This construction ensures that the portfolio is risk-free by balancing the position in the underlying asset with the short position in the option.

For the portfolio P_t to be risk-free, its future value must be identical in both the up and down states. This condition leads to:

$$\Delta S_u - V_u = \Delta S_d - V_d$$

Solving for the hedging ratio Δ associated with the number of units of the underlying asset yields:

$$\Delta = \frac{V_u - V_d}{S_u - S_d}$$

This calculation ensures that the portfolio replicates the option's payoff in both possible future

2.2.2 Insights

The following figure illustrates the evolution of computation time as the number of steps increases in the multi-step binomial tree model.

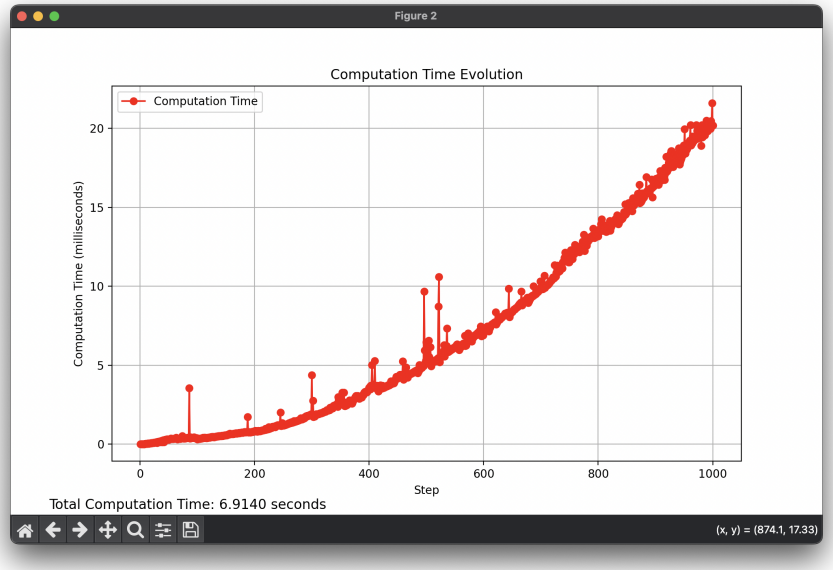

Figure 3: Computation time of each step in the binomial tree model (1000 steps)

The observed quadratic trend in computation time suggests that the model's complexity scales poorly with large steps. Optimization strategies such as dynamic programming, parallel processing, and memory-efficient algorithms should be explored to improve performance. If we look at the code snippet above, the time complexity is dominated by the nested loops:

Outer loop: Iterates over steps from n to 0 (total $n+1$ steps). Inner loop: For each step s, iterates i from 0 to s (total $s+1$ iterations per step).

Per-node operations:
For each node (step, i), `calculateOptionValue` computes the option value using:

$$\text{Value} = \frac{q \cdot \text{Payoff}_{\text{Up}} + (1-q) \cdot \text{Payoff}_{\text{Down}}}{1+r}$$

This involves constant-time operations (addition, multiplication, division), so it contributes $O(1)$ per node.
This simplifies to a time complexity of:

$$O(n^2)$$

Per-Node Operations For each node (s, i):

- Computes stock price: initialPrice $\times$ (upFactor)^i $\times$ (downFactor)$^{s-i}$ (constant time).
- Computes option payoff (constant-time comparison).
- Discounts future values via `calculateOptionValue` where for each node, `calculateOptionValue` computes the option value using:

$$\text{Value} = \frac{p \cdot \text{Payoff}_{\text{Up}} + (1-p) \cdot \text{Payoff}_{\text{Down}}}{1+r}$$

This involves constant-time operations, so it contributes $O(1)$ per node.