POLITECNICO DI TORINO

Master's Degree in Mathematical Engineering

Master's Degree Thesis

Dynamic Hedging of exotic derivatives via Stochastic Optimization: development and benchmarking with Delta Hedging and Deep Hedging

Supervisors

Prof. Paolo BRANDIMARTE

Prof. Edoardo FADDA

Ph.D. Giovanni AMICI

Candidate

Alessandro ORAME

March 2025

Summary

In the context of financial markets, a constant element is the presence of risks, which can make the management of financial instruments challenging. The complexity may increase in the case of exotic derivatives, whose value may depend on a variety of underlying variables and risk factors. In this framework, the present work develops a strategy for the dynamic hedging of exotic derivatives through stochastic optimization.

The main idea behind a hedging problem is to optimize the management of a hedging portfolio designed to offset potential future liabilities arising from a position in the exotic derivative. The term 'dynamic', instead, refers to a multi-stage decision process where multiple decisions can be made over time. To deal with a dynamic hedging problem, the implemented strategy lies within the stochastic optimization framework, formulating, at each decision stage, an optimization problem which reflects the underlying idea of an Asset-Liability-Management problem. The optimization model accounts for a set of stochastic variables and leverages scenario trees for a discrete representation of possible future outcomes. To accurately approximate the space of future scenarios, two methods to simulate financial instrument dynamics are employed: the Geometric Brownian Motion simulation and the Moment Matching method.

After an initial development phase, the proposed approach is evaluated in terms of effectiveness and efficiency: Monte Carlo simulation is leveraged to test the implemented strategy over a set of simulated paths for the financial market dynamics. Lastly, a comparative analysis is performed to assess the effectiveness of the proposed strategy compared to traditional methods, such as Delta Hedging, and more recent approaches, such as Deep Hedging.

Table of Contents

List of Tables

List of Figures

Acronyms

ALM
Asset-Liability management

BM
Brownian Motion

bf
Branching factors

cdf
Cumulative density function

GBM
Geometric Brownian Motion

LP
Linear Programming

LSTM
Long short term memory (neural network)

MC
Monte Carlo (simulation)

MM
Moment Matching

NN
Neural network

P&L

Profit and Loss

QP

Quadratic Programming

RNN

Recurrent neural network

SDE

Stochastic differential equation

Chapter 1

Foundations and context of hedging

Managing risks in financial markets is a nontrivial challenge, particularly when dealing with complex financial instruments. The idea of a hedging problem is to minimize, through the application of advanced mathematical techniques, the risk exposure of financial market participants when trading instruments, thus reducing potential future losses. The work described in this thesis focuses on hedging exotic financial derivatives through a multi-stage stochastic optimization approach, providing a hedging strategy which relies on an optimized portfolio management under uncertainty.

In this chapter, the fundamentals to understand the hedging problem are presented. In a first phase, the concept of hedging is introduced, examining the motivations to formulate a hedging problem and the approach to its resolution; some well-known hedging strategies in the literature are then presented, namely Delta Hedging and Deep Hedging: these strategies will be used in further chapters to benchmark the hedging strategy through Stochastic Optimization proposed by this work.

For a better comprehension of the general framework in which the hedging problem is formulated, following an introductory section, the financial framework is established by describing the market setup along with its underlying dynamics, as well as by defining the financial instruments considered in the hedging strategy, along with their characteristics and mathematical formulations.

1.1 Introduction to the hedging problem

As mentioned above, when facing financial markets one recurring feature is the presence of risk, which can manifest in various forms: potential losses may arise by unfavorable price movements of any assets, incurring what is called market risk; an asset may become difficult to trade due to bid-ask spread or low trading volumes, leading to liquidity risk, or again a counterpart in a financial contract may fail to meet its obligations, facing credit risk or counterpart risk, and many other different risks can arise in financial markets. In the context of financial derivatives trading, managing these risks is crucial and investors may need advanced models and strategies to hedge against them.

In this work, the following risk scenario is considered: a short position in an exotic derivative is taken, which may result in the obligation to close the financial contract under adverse conditions if the derivative ends up in-the-money at maturity. Thus, holding the short position implies dealing with the risk of possible future losses depending on the market scenario that will occur. The framework established in this work only focuses on option contracts, a specific set of financial derivatives better detailed in section 1.2. The party holding the short position in an option contract can possibly face unbounded losses (in extreme scenarios), in contrast to the bounded loss of a long position; in addition, the short party cannot avoid his obligation when the option is exercised by the counterpart: due to this asymmetry in the contract, the short position is charged with an amount known as the option premium. At this point, the hedging strategy comes into play: the option premium received from the counterpart can be immediately reinvested to construct a hedging portfolio; the latter consists of a portfolio of different assets, denoted by hedging instruments, with the aim of replicating the exotic option's payoff. Once a replicating portfolio is constructed, the hedger is provided with an instrument which ensures that, with high probability, the future loss deriving from the target asset is offset by the portfolio: this concept is related to the minimization of the hedging error, defined as the difference, at the target asset expiration date, between the target payoff and the hedging portfolio value. The idea is that, by differentiating our wealth in an optimized portfolio, the risk and uncertainty of the unhedged position (one in which no hedging strategy is applied, leaving its value to depend entirely on the target payoff) is mitigated, especially in terms of variance of potential losses.

The strategy presented above applies both for static and dynamic approaches: the difference between these two is better detailed in the following subsection.

1.1.1 The concept of dynamic hedging

Static and dynamic hedging are two distinct approaches to the hedging problem: their difference lies in the number of stages in which the hedger makes decisions.

Static Hedging involves a single decision stage in which the hedger optimizes the strategy, constructing the hedging portfolio once and maintaining it unvaried until the end of the hedging horizon, when its value is used to offset liabilities which may be encountered. However, particularly for long-term hedging horizons, this approach may not be effective in minimizing risks, due to market fluctuations which might occur after the portfolio construction at the initial stage, leaving the hedger exposed to unexpected risks.

Dynamic Hedging, on the other hand, involves a sequence of consecutive decision stages over the hedging horizon: at a first stage, the hedging portfolio is constructed, while for subsequent stages its composition is adjusted according to the financial market realizations; this strategy allows for a more effective hedging since the portfolio is rebalanced periodically and adapts to the new market conditions, leveraging the additional knowledge acquired along the path up to the current decision stage. Although dynamic hedging provides a more flexible response to market fluctuations, it also requires a more active management of the strategy, as well as addressing market frictions such as transaction costs, which may erode the wealth invested in the portfolio if portfolio rebalancing is too frequent.

This thesis focuses solely on the implementation of a dynamic hedging strategy due to its superior effectiveness, despite its higher complexity.

1.1.2 Hedging strategies in the literature

Many hedging strategies have been studied over the years, also with quite different methodologies and applications, aiming to mitigate different financial risks: most strategies are developed to hedge against assets' volatility, but strategies for interest-rate movement, credit risk, event and liquidity risks and others exist too. In light of this, a brief overview of the key hedging strategies proposed in the literature is given in this section.

A first classification of hedging strategies was already given in the previous section, concerning the difference between static and dynamic hedging. A further distinction among hedging strategies relies on the approach used to derive a hedging policy: indeed, hedging strategies can be either model-based or data-driven. While the formers rely on financial market assumptions, such as the Black-Scholes framework, and derive analytical method to hedge a specific risk, data-driven approaches,

like machine learning or reinforcement learning, leverage the knowledge contained in historical or simulated data to learn a hedging policy that can be applied to new unseen data. One approach for each category is presented in the following sections and later used in subsequent chapters as a benchmark for our stochastic programming method.

Delta Hedging

The Delta Hedging is one of the most well-known hedging strategies studied in the financial literature; it belongs to a set of hedging approaches related to the concept of **Greeks**, some risk measures which describe the variation of an option contract value with respect to some risk factors. In particular, the delta (Δ) of an option is defined as the sensitivity of the option value to changes in the underlying value, or better the rate of variation of the option price with respect to the change in its underlying value. Denoting by P_t and S_t the option and the underlying prices at time t, we have:

$$\Delta_t = \frac{\partial P_t(S_t)}{\partial S_t}. \tag{1.1}$$

Assuming to have a short position in an option contract, Delta Hedging is a risk management strategy aimed at maintaining a **delta-neutral** portfolio, buying or selling an amount of the underlying equal to the value of delta; in this way, for small changes in the underlying price, there would be no changes in the portfolio value as the option and the underlying's changes in value offset each other. The strategy may also include a risk-free asset to cover portfolio costs and guarantee a self-financing condition. M.Villaverde formalizes in the paper "*Hedging European and Barrier options using stochastic optimization*" [1] a Delta Hedging strategy to hedge against a short position in an option with a portfolio composed of the underlying stock and a risk-free asset; denoting by B_t the value of the risk-free asset, S_t the value of the stock, $X_t = (X_t^0,\ X_t^1)$ the vector of assets' holdings in the portfolio, where X_t^0 refers to the risk-free asset and X_t^1 to the stock, and an initial wealth for the portfolio construction W_0, in absence of transaction costs, Villaverde's strategy for each decision stage would be:

$$X_0^1 = \Delta_0, \quad X_0^0 = \frac{W_0 - X_0^1 S_0}{B_0},$$

$$X_t^1 = \Delta_t, \quad X_t^0 = X_{t-1}^0 + \frac{(X_{t-1}^1 - X_t^1)S_t}{B_t}, \quad \forall\ t = 1, \dots, T-1$$

and

$$X_T^1 = \Delta_{T-1}, \quad X_T^0 = X_{T-1}^0.$$

Just for reference, moving away from the delta but within the Greeks, we can mention gamma Γ, which is a measure of delta sensitivity with respect to the

underlying price, but also measures against other risk factors, such as theta Θ, (option price sensitivity to time), rho ρ, (sensitivity to interest rate r), and vega ν (sensitivity to volatility σ); for each Greek (and also for combinations of them) a hedging strategy can be constructed similarly to the Delta Hedging.

Deep hedging

In contrast to traditional hedging methods, the growing interest in machine learning led to the development of machine learning algorithms applicable to the financial hedging problem. The term Deep Hedging refers to the application of neural networks (NN) to learn policies for portfolio construction and rebalancing aiming at hedging a position in a target asset. Unlike most of other hedging strategies, which are model-based, Deep Hedging is a data-driven, learning the optimal hedging strategy by training the network over a large number of simulated scenarios.

Just to provide the reader with a first introduction to the topic, neural networks are machine learning models widely used for their ability to learn complex and non-linear relationships among data. They features a set of consecutive layers of connected nodes (each with assigned weights) through which data is processed with linear and non-linear functions, starting from the input layer, proceeding through the intermediate ones (the hidden layers) up to the final one, the output layer. In particular, a weight matrix W_i and a bias vector b_i are assigned to each node i, by means of which the following linear transformation is applied: $X_i = W_i \cdot \boldsymbol{x}_i + b_i$, where $\boldsymbol{x}_i$ is a matrix containing the outputs of the nodes from the previous layer. To the term X_i is then applied a non-linear function, among which the most well-known are the sigmoid function and the hyperbolic one; the result of these two steps is then passed to nodes of the following layer.

Each sample of the data is processed through the NN resulting in an output vector on which a performance measure is computed and the nodes' weights and biases are then adjusted in order to optimize the performance measure over the whole available dataset. Figure 1.1 shows an example of the structure of a neural network.

In the hedging framework, given a portfolio with multiple hedging assets, the neural network takes as input the current state of the financial market and outputs the optimal holdings of the hedging instruments to be held in the portfolio. The ideal network for a multi-stage decision process turns out to be a recurrent neural network (RNN), a specific type of NN where the input and the output are sequences of dependent values rather than vectors: each layer of a RNN takes as input not only the output of the previous layer but also its own output of the previous time step, making the network well-suited for problems where there is temporal

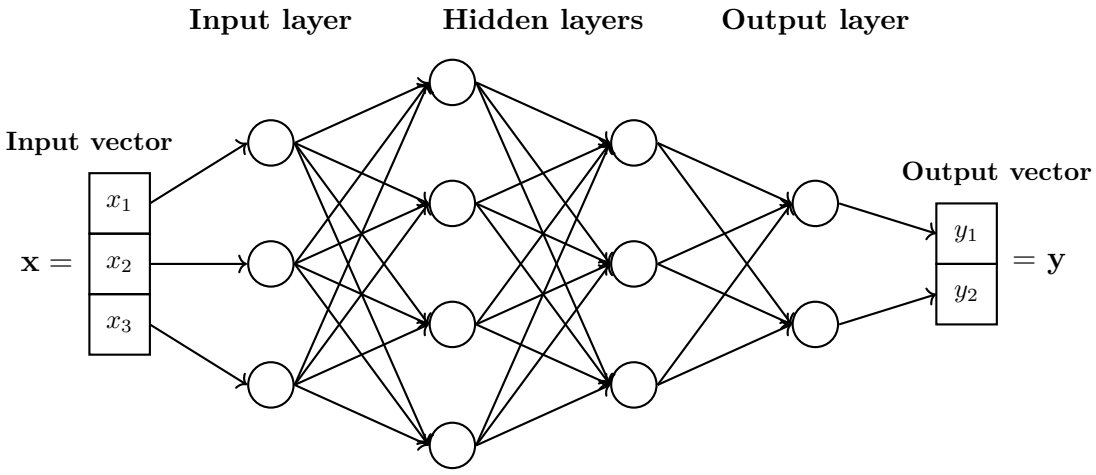


Figure 1.1: Example of a (fully connected feed forward) neural network with 2 hidden layers.

dependency in the data, as in the case of the consecutive decisions made in dynamic hedging.

For the scope of this thesis, an example of Deep Hedging algorithm existing in the literature is taken as a benchmark for our stochastic optimization model; in particular, it is taken into account the long-short term memory RNN described by Alexandre Carbonneau in the article "*Deep Hedging of long-term financial derivatives*" [2]. For further details on the implementation, refer to chapter 6 (or directly to the reference).

1.2 Market setup

This thesis develops a stochastic programming method applied to the risk management in financial applications. Hence, it is essential to provide some fundamental notions that will be used throughout the work for the full comprehension of the topic. For this reason, this section outlines the financial market setup which serves as the background framework for the hedging problem proposed. In particular, the financial instruments available in the market are described, with a specific focus on their characteristics and behavior.

The financial market is in discrete time, with a finite time horizon of length T, equal to the target asset maturity, divided into K equal-length time intervals Δt

by the set of observation dates $\{t_k\}_{k=0}^{K}$, also denoted as time instants. In formula:

$$t_k = k\Delta t \quad \forall k \in \{0, \dots, K\}, \quad \text{where} \quad \Delta t = \frac{T}{K}.$$

Recall that hedging a short position in a financial derivative implies the construction of a hedging portfolio. Contrary to what one might expect, also exotic derivatives with complex payoff functions can be hedged using a finite number of simple financial instruments: this strategy is based on the principle that exotic options can be decomposed or replicated, at least partially, with combinations of simpler assets. In light of this, the financial market will take into account a finite set of basic financial assets for the construction of the hedging portfolio; specifically, the following instruments will be considered:

- a **bank account**, denoted for the sake of simplicity by *Cash position* B, which plays the role of the riskless asset, earning a risk-free rate of return;
- a set of **stocks** S_i for $i \in \{1, \dots, m\}$, with each stock representing a risky asset subject to stochastic price dynamics;
- vanilla **European options** (including both call and put options), which are basic financial contracts written on the available stocks.

Since this work addresses not only vanilla options but also exotic ones, a brief overview of financial derivatives should be provided.

Financial derivatives, or contingent claims, are instruments whose value depends on the value of one or more other assets, called underlying assets; they play a key role in financial markets, especially when dealing with hedging against risks or speculating on future market movements. Among the most well-known derivatives are listed options, futures and forward contracts. For the purpose of this work, the focus will be on option contracts only.

European options

European options are financial contracts which give the holder the right, but not the obligation, to buy or sell an underlying asset at a predetermined price, known as the strike price K, and at a predetermined date, the option maturity T. In contrast to the option holder, there is the option writer, who writes and sells the contract on the market and who has the obligation to buy or sell the asset if the holder exercises the right. Due to the asymmetry in the obligations of the two counterparts, the writer requires a premium for the risk he assumes. For option contracts, the writer is said to have a short position, while the holder has a long position. There are two types of European options, classified by the payoff at maturity: European call option and European put option.

European call options. A European call option is a contract which gives the holder the right to buy the underlying asset at the strike price K at maturity T. For a call written on a stock whose price at time t is denoted by S_t, the payoff function for the long position is defined by:

$$\text{Payoff}_{\text{long}} = \max(S_T - K, 0).$$

Given that the holder pays a premium p to enter the contract, the profit of the long position as a function of the underlying price at maturity is:

$$\text{Profit}_{\text{long}} = \max(S_T - K, 0) - p.$$

Intuitively, the payoff and profit functions for the short position on a call option are given by:

$$\text{Payoff}_{\text{short}} = -\max(S_T - K, 0),$$

$$\text{Profit}_{\text{long}} = -\max(S_T - K, 0) + p.$$

Figure 1.2 and Figure 1.3 show profits and payoffs of the two parties in a European call option contract, respectively for the long and the short position.

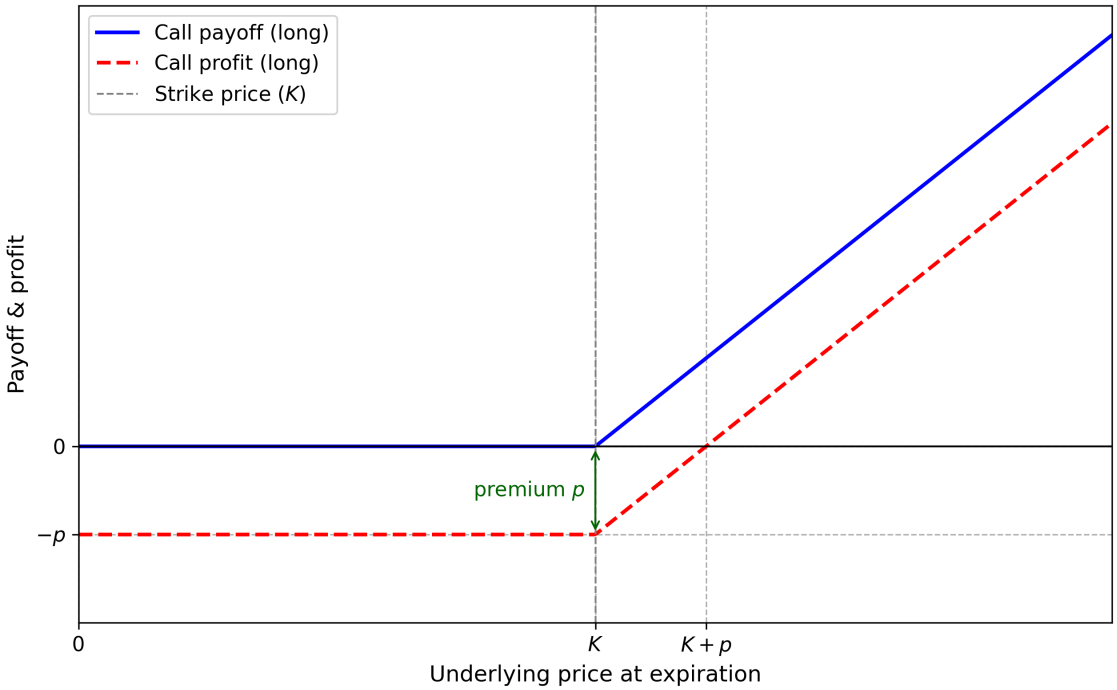


Figure 1.2: Payoff and profit for a long position in a European call option.

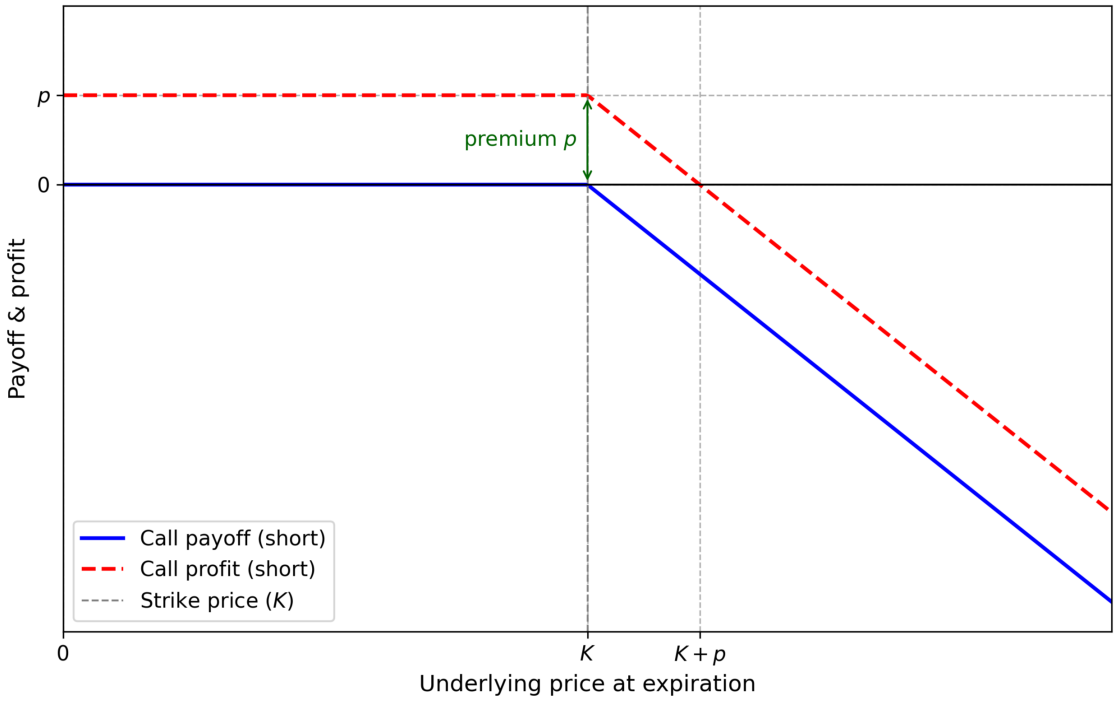


Figure 1.3: Payoff and profit for a short position in an European call option.

European put options. Opposed to a European call option, the European put option is a contract that gives the holder the right to sell the underlying asset at the strike price K at maturity T. Thus, payoff and profit formulas for the long position in an European put option are:

$$\text{Payoff}_{\text{long}} = \max\left(K - S_T, 0\right),$$

$$\text{Profit}_{\text{long}} = \max\left(K - S_T, 0\right) - p,$$

while formulas for the short position in an European put are:

$$\text{Payoff}_{\text{short}} = -\max\left(K - S_T, 0\right),$$

$$\text{Profit}_{\text{short}} = -\max\left(K - S_T, 0\right) + p.$$

Figure 1.4 and Figure 1.5 report the graphs of these functions, respectively for the long and the short position.

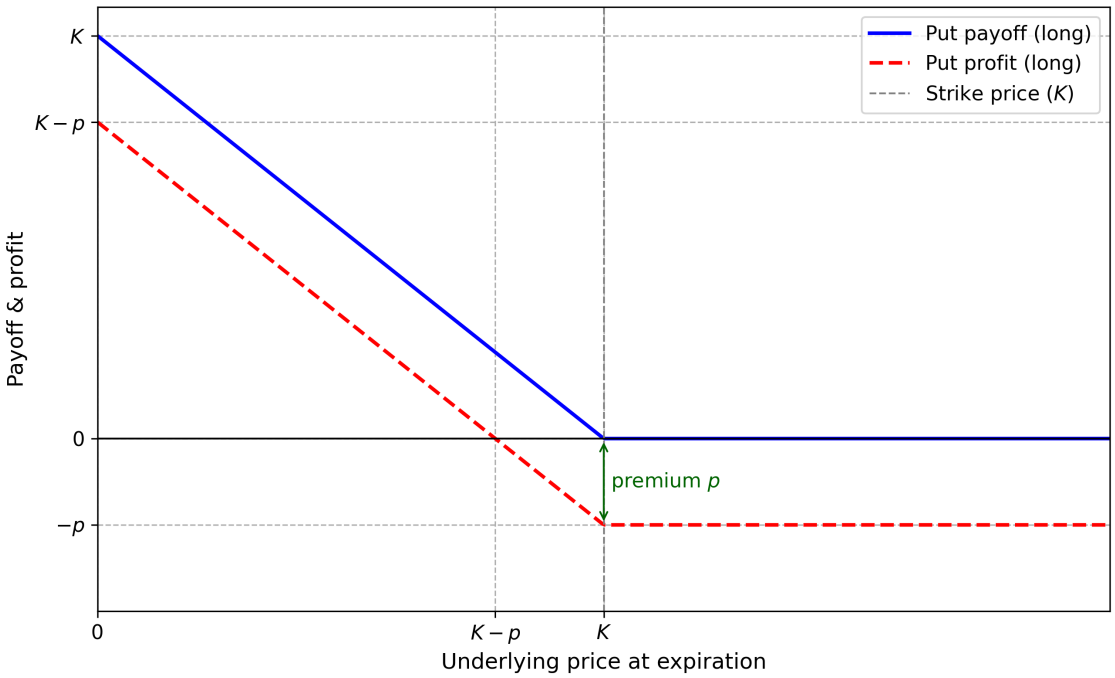


Figure 1.4: Payoff and profit for a long position in a European put option.

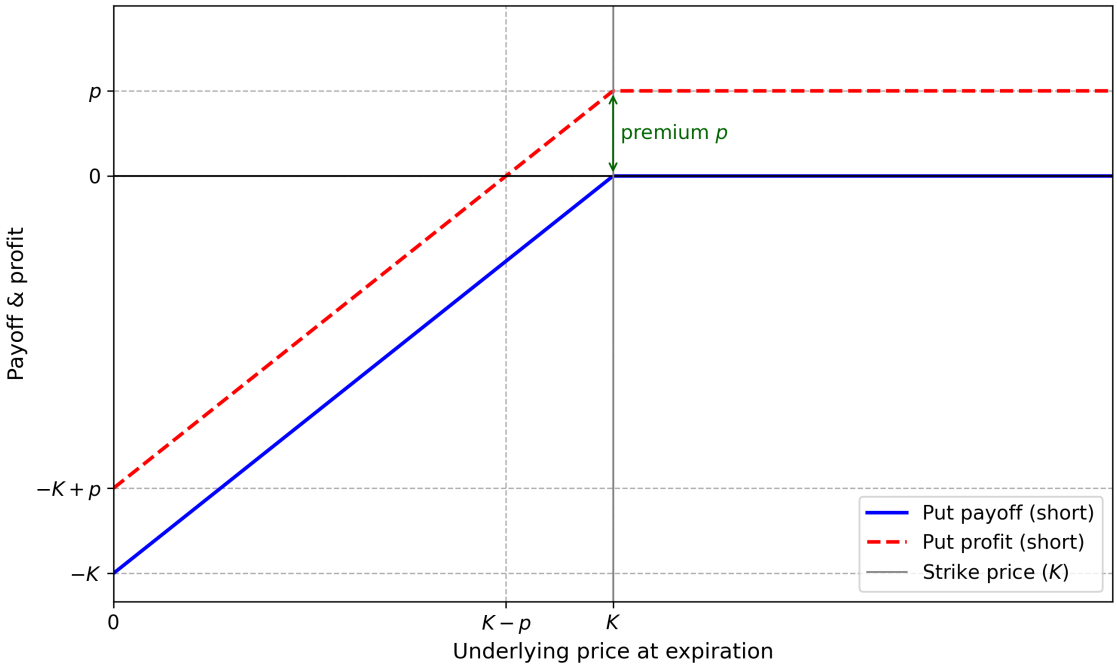


Figure 1.5: Payoff and profit for a short position in an European put option.

Asian options

Asian options are financial derivatives similar to European options, except that they can be written on multiple underlying assets and their payoff at maturity depends on the average value of the underlyings over time. Asian options can be both European and American style[1], but we focus on European style options only, thus options that can be exercised exclusively at expiration date T. For a long Asian option, the payoff formulas are:

$$\text{Payoff}_{\text{call}} = \max\left(\bar{S} - K, 0\right),$$

$$\text{Payoff}_{\text{put}} = \max\left(K - \bar{S}, 0\right),$$

where $\bar{S}$ is an average over a finite number of time instants during the contract lifetime, that is:

$$\bar{S} = \frac{1}{n}\sum_{k=1}^{n}\left(\frac{1}{m}\sum_{i=1}^{m} S_{i,t_k}\right),$$

with n representing the number of time instants and m the number of underlying stocks. All remaining formulas for Asian options (calls and puts, in both short and long positions) can be derived trivially by following the logic of the European option formulas.

1.3 Market dynamics

In this section, the focus shifts to the dynamics that model the evolution of the financial market. In particular, the stochastic processes that govern asset price paths are explored, with particular attention given to the key role of Brownian Motion and Geometric Brownian Motion in the context of this thesis, as well as their relevance in the Black–Scholes–Merton pricing framework. For reference, and without loss of generality, a stochastic process can be defined as a family of time-dependent random variables $\{X_t\}_{t\in T}$.

The discussion of this section will lay the basis for the simulations of the market evolution developed in later chapters.

[1]American options are contracts in which the holder has the right to buy or sell the asset at strike price K at any time, or at specific dates, during the contract lifetime, not necessarily only at maturity.

1.3.1 Brownian motion

Brownian motion is a continuous-time stochastic process which is well-known in academic literature and widely used in financial applications.
Mathematically, a **standard Brownian motion** $\{W_t\}_{t\geq 0}$ is nothing more than the Wiener process; thus, before entering into details, it is appropriate to make some reference to the latter.

Wiener process and drifted Brownian Motion

The Wiener process $\{W_t\}_{t\in T}$ is a continuous-time stochastic process with the following properties:

- $W_0 = 0$;
- independent and normally distributed increments; in particular:
$$W_{t+s} - W_t \sim \mathcal{N}(0, s);$$
- continuous sample path, that is $W(t)$ is a continuous function of t.

As already mentioned, the standard Brownian motion process is another notation for the Wiener process W_t. The process can be generalized to the **drifted Brownian motion** $\{B_t\}_{t\in T}$, defined as

$$B_t = x_0 + \mu t + \sigma W_t,$$

where $B_0 = x_0$ is the initial value, μ is the *drift term*, σ is the *diffusion term* and W_t is the Wiener process. The stochastic differential equation (SDE) under the drifted Brownian motion is:

$$dB_t = \mu dt + \sigma dW_t.$$

This process is typically associated with the log-transformation of the stocks price process: given a stock S_t and an estimate of the its expected return and volatility (μ and σ respectively), it is generally said that:

$$d(\ln(S_t)) = \left(\mu - \frac{\sigma^2}{2}\right) dt + \sigma dW_t,$$

that is the equation of a drifted Brownian motion with drift term $\left(\mu - \frac{\sigma^2}{2}\right)$ and diffusion term σ; thus we have that, for any $s < t$,

$$\ln(S_t) \sim \mathcal{N}\left(\ln(S_s) + \left(\mu - \frac{\sigma^2}{2}\right)(t-s), \sigma^2(t-s)\right),$$

implying that S_t follows a log-normal distribution.

A further generalization is the definition of Geometric Brownian motion, which is in fact a common choice for the stocks' price process.

1.3.2 Geometric Brownian motion

The stock price S_t follows the stochastic differential equation of the geometric Brownian motion:

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{1.2}$$

where:

- μ is the expected return of the stock;
- σ is the standard deviation of the stock's returns;
- W_t is the Wiener process.

The drift term of equation (1.2) is given by μS_t, while the diffusion term is given by σS_t. Without entering into mathematical details, the solution to the SDE is given by:

$$S_t = S_0 \cdot e^{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t}. \tag{1.3}$$

The Brownian Motion (and Geometric Brownian Motion) are not only important in modeling stock price dynamics, but they also play a key role in the Black-Scholes-Merton pricing formula for European-style options.

1.3.3 Black-Scholes-Merton pricing formulas

Black-Scholes-Merton formulas are among the most well-know pricing formulas in financial applications. Despite their popularity, they rely on a set of assumptions that simplify the complexity of financial markets and allow for the derivation of a closed-form solution for the price of vanilla European-style options. The main assumptions are the absence of arbitrage opportunities and frictions[2] in the market and log-normal asset price distribution (the price of the underlying asset follows a Geometric Brownian Motion, implying that its log-return is normally distributed with constant volatility.).

Given these conditions, it is possible to derive European option prices. In particular, the European call option price is given by:

$$C_t = S_t \cdot N(d_1) - K \cdot e^{-r(T-t)} \cdot N(d_2), \tag{1.4}$$

while the European put option price is:

$$P_t = K \cdot e^{-r(T-t)} \cdot N(-d_2) - S_t \cdot N(-d_1), \tag{1.5}$$

[2]transaction costs or similar.

where K is the option strike, S_t is the underlying price at time t, $N(\cdot)$ is the cumulative distribution function (cdf) of a standard normal and the formulas for d_1 and d_2 are:

$$d_1 = \frac{\ln\left(\frac{S_t}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}},$$

$$d_2 = d_1 - \sigma\sqrt{T-t} = \frac{\ln\left(\frac{S_t}{K}\right) + \left(r - \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}.$$

Given the Black-Scholes-Merton formulas for the price of vanilla European options, it is trivial to compute their Delta applying Equation 1.1.
For a European call option:

$$\Delta_{\text{call}} = N(d_1),$$

while for a European put option the delta is given by:

$$\Delta_{\text{put}} = N(d_1) - 1.$$

Pricing Asian options

Black-Scholes-Merton formulas are specifically for vanilla European-style options; European-style Asian options cannot be priced using the Black-Scholes-Merton model. Asian options have payoffs that depend on the average of the underlying assets prices over time, making their pricing more complex. In the context of this thesis, Monte Carlo simulation is used to price these options: a high number of stock price paths is simulated[3] to estimate the expected payoff of the Asian option:

$$\mathbb{E}[\text{payoff}] \approx \frac{1}{|\Omega|}\sum_{\omega\in\Omega} \text{payoff}(\omega);$$

the price of the Asian option is then computed as the discounted expected value of the payoff:

$$\text{Asian option price} = e^{-rT}\mathbb{E}[\text{payoff}].$$

[3]Note that, from a theoretical point of view, the pricing framework should be done assuming a risk neutral (hypothetical) world, that is: the simulation of stock prices evolution for the pricing of Asian option is performed assuming that the expected return of the stocks is equal to the risk-free rate r.

Chapter 2

Multi-stage Stochastic Optimization

The dynamic hedging problem can be viewed as a sequential decision-making problem where the hedger periodically rebalances the hedging portfolio over the time horizon, aiming to minimize the risk exposure related to possible future liabilities. However, the process is subject to a set of stochastic variables whose realizations are unknown in advance; therefore, decisions must be robust against a wide set of possible future scenarios. The primary source of stochasticity in financial applications, which will be the focus of our attention, is the evolution of asset prices.

In this chapter, the methodology applied for the development and implementation of the hedging strategy is presented. First, the mathematical formulation of multi-stage stochastic decision models is introduced to better clarify how sequential decision-making under uncertainty can be formalized using stochastic optimization techniques.
Just after that, scenario trees will be introduced for a better understanding of how the hedging problem will be faced in following chapters. Their role in modeling the uncertainty about future market outcomes will be detailed, along with the underlying dynamics modeling the generation of values in their discrete approximation of market evolution. In this context, the Geometric Brownian Motion previously introduced will play a key role, together with a method known as Moment Matching. Lastly, some discussions will be made about different approaches to ensure the absence of arbitrage opportunities in the simulated paths of scenario trees.

2.1 Fundamentals of Stochastic Optimization

Stochastic Optimization is a framework of decision-making problems under uncertainty: indeed, it is a family of optimization models whose objective function and constraints may depend both on decision variables and on realizations of stochastic variables that are encountered in the environment. For the sake of simplicity, it is assumed that there is no stochasticity in the constraints but only in the objective function (that is the case of our problem implementation).

Without loss of generality, a one-stage stochastic optimization problem can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \mathbb{E}_\Omega \left[f(\boldsymbol{x}, \boldsymbol{\xi}) \right] \\ \text{s.t} \quad & h(\boldsymbol{x}) = 0 \\ & g(\boldsymbol{x}) \leq 0 \end{aligned}$$

where $\boldsymbol{x}$ is the vector of decision variables and $\boldsymbol{\xi}(\omega)$ is the vector of stochastic variables which depend on the scenario $\omega \in \Omega$.

A multi-stage stochastic optimization model is a SO problem in which different decision-making stages follow each other, introducing a temporal dependence in the model: at each stage, the decision is based on the information available up to the current time. A finite time-horizon multi-stage SO problem can be formulated as:

$$\begin{aligned} \min_{\{\boldsymbol{x}_t\}_{t=1\ldots,T}} \quad & \mathbb{E}_\Omega \left[\sum_{t=1}^{T} f_t(\boldsymbol{x}_t, \boldsymbol{\xi}_t) \right] \\ \text{s.t} \quad & \boldsymbol{x}_t \in \mathcal{X}_t(\mathcal{F}_{t-1}) \quad \forall\, t = 1, \ldots, T \end{aligned} \tag{2.1}$$

where T is the number of decision stages, $\mathcal{F}_t$ is the σ-algebra[1] representing the information available up to time t and $\mathcal{X}_t(\mathcal{F}_{t-1})$ represents the feasibility set to which x_t belongs, indicating that decision variables may be constrained according to the previous decision made and to the realizations of the stochastic variables observed up to the current time t.

Concerning the hedging problem, the latter is in fact formulated as a multi-stage stochastic optimization problem, despite some considerations which result in a simplified version of Problem (2.1):

[1] Briefly, $\mathcal{F}_t$ is the set containing all the events happened and decisions made up to time t; it can be seen as the history experienced before reaching time t.

- although decision stages follow each other along the whole time horizon, the objective of the hedging boils down to minimizing the hedging error at the target maturity date; thus, the only contribution to the objective function is given by

  $$f_T(\mathcal{F}_T),$$

  where f_T will represent a function computing the hedging error and the presence of $\mathcal{F}_T$ indicates that the contribution to the objective function is given by the whole history of decisions and realizations of stochastic variables up to time T;

- in order to have a consistent representation of the uncertainty about the stochastic variables future realizations and to estimate the expectation in Problem (2.1), Multi-stage Stochastic Optimization problems are often supported by the use of scenario trees, which help to model the evolution of stochastic variables over time and to approximate the probability distribution of possible future scenarios by providing a discretized version of it. This discretization is achieved by dividing, at each stage, the possible future market states into discrete branches, each representing a potential realization of stochastic variables.

Considering a discrete set $\tilde{\Omega}$ of possible future scenarios with cardinality $|\tilde{\Omega}| = n$ (equipped with a probability density function π) given by a scenario tree, the hedging problem follows the formulation:

$$\begin{aligned} \min_{\{\boldsymbol{x}_t\}_{t=1,\dots,T}} \quad & \frac{1}{n}\sum_{i=1}^{n}\Big(f_T(\mathcal{F}_T^{(i)})\cdot\pi(\tilde{\omega}_i)\Big) \\ \text{s.t} \quad & \boldsymbol{x}_t \in \mathcal{X}_t(\mathcal{F}_{t-1}) \quad \forall\, t = 1,\dots,T \end{aligned}$$

where $\mathcal{F}_T^{(i)}$ includes all the realization of stochastic variables which led to the scenario $\tilde{\omega}_i$ and all the decisions made along the path. In this re-formulation of Problem (2.1), the expectation $\mathbb{E}_\Omega[\cdot]$ is replaced by the sample mean over the set of simulated scenarios in the scenario tree. In order to ensure an effective and accurate optimization problem, it is crucial to generate scenario trees in a careful way, taking into account the expected dynamics of the random variables involved.

In light of this, the following section is dedicated to the description of scenario tree generation in financial applications, with a particular focus on the dynamics implemented for the purpose of this thesis.

2.2 Scenario Tree Generation

As evidenced by M.Villaverde,

> "The effectiveness of the stochastic optimization based hedging strategy depends crucially on an accurate representation of the uncertainty and thus on an accurate scenario tree" [1].

Scenario trees play a key role in stochastic optimization problems as they are the means through which uncertainty is modeled over time: indeed, they are a graphical representation of some possible future outcomes of any uncertain parameters in the model.

Scenario trees have three main features:

- branching steps: the number of times the tree branches, i.e. the number of stages in the decision process;
- length of time steps: the time interval between one node and its children. Time steps are strictly related to scenario trees of time-dependent models, that is the case of stochastic optimization models where there is a temporal structure in the decision process;
- branching factors: the number of children nodes generated from each node in the tree. The branching factors can be homogeneous (when the branching factor is constant for all nodes), homogeneous by time step (when the branching factor is constant within each time step but may vary across time steps) and inhomogeneous (different for each node in the tree).

The number of final scenarios generated by the tree in case of branching factors homogeneous by time step is trivially computed as the product of branching factors (e.g. with branching factors $[30, 10, 4]$ the number of scenarios is $30 \cdot 10 \cdot 4 = 1200$). Figure 2.1 shows a graphical example of a scenario tree.

An important aspect related to scenario trees concerns the underlying dynamics determining the evolution of generated values for stochastic variables. In the hedging framework, the stochastic variables correspond to assets' prices (in particular, stocks' prices) and the scenario tree should capture appropriately the properties and the assumptions on the price dynamics of these assets.
In the implementation of the stochastic optimization hedging problem different approaches to scenario tree values generation are considered: the Geometric Brownian Motion and the Moment Matching approach; while the first was already mentioned before, the latter is briefly introduced in the following paragraph.

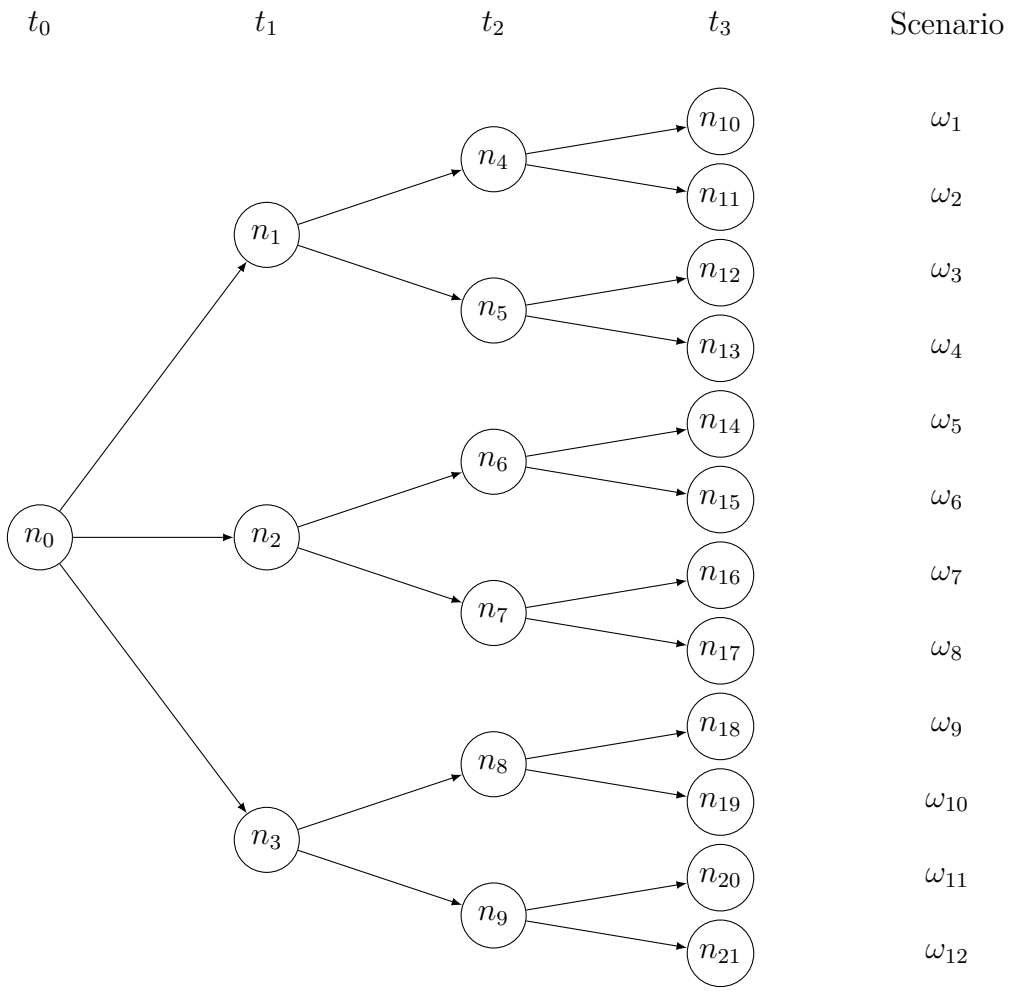


Figure 2.1: Example of scenario tree with 3 branching steps and branching factors $[3, 2, 2]$, with a total of 12 scenarios.

Moment Matching. The core idea of Moment Matching is to simulate stock prices from a distribution that best approximates the distribution of some historical data, where the approximation is achieved by matching moments and properties of the two distributions (such as first and second moments, variance, correlation, skewness, kurtosis and more); for the application of Moment Matching in this work, the generation of values for a scenario tree is achieved by solving an optimization problem where the properties of the generated scenario tree are forced to match the correspondent properties of the historical data distribution.
Given a hint on the moment matching technique, the following sections provide a brief overview of the implemented dynamics in scenario tree generation.

Stocks Parameters Estimation. To better describe the simulation of stock dynamics in scenario tree generation, this paragraph provides a brief reference to the estimation of stock dynamics parameters.

By means of a set of available historical data, daily logarithmic returns of each stock $j = 1, \ldots, m$ are computed as follows:

$$Y_j(t) = \ln\left(\frac{S_j(t)}{S_j(t-1)}\right),$$

where t denotes the time index in days.

The statistical moments and properties of stock dynamics are then inferred based on these log-returns; the set of estimated parameters includes:

- drift $\hat{\boldsymbol{\mu}} = [\hat{\mu}_1, \ldots, \hat{\mu}_m]$,
- volatility $\hat{\boldsymbol{\sigma}} = [\hat{\sigma}_1, \ldots, \hat{\sigma}_m]$,
- the correlation matrix $\hat{\boldsymbol{\rho}}$, with each entry $\hat{\rho}_{j,k}$ corresponding to the correlation between log-returns of stocks j and k;
- skewness $\hat{\boldsymbol{\gamma}} = [\hat{\gamma}_1, \ldots, \hat{\gamma}_m]$
- kurtosis $\hat{\boldsymbol{\nu}} = [\hat{\nu}_1, \ldots, \hat{\nu}_m]$.

After a first estimation from daily log-returns, parameters are annualized.

One aspect of particular interest concerns the drift $\hat{\boldsymbol{\mu}}$, which is adjusted by adding the term $\hat{\boldsymbol{\sigma}}^2/2$, that is:

$$\hat{\mu}'_j = \hat{\mu}_j + \frac{\hat{\sigma}_j^2}{2} \quad \forall j = 1, \ldots, m.$$

This choice is motivated by the fact that the drift is estimated, in a first phase, directly from log-returns; however, to exploit GBM formulas with a coherent drift term, this adjustment is necessary, leading to the term $\hat{\boldsymbol{\mu}}'$ which corresponds to the expected stocks return.

From now on, the notation $\hat{\boldsymbol{\mu}}$ will refer to the adjusted drift term which incorporates the correction $\hat{\boldsymbol{\sigma}}^2/2$.

2.2.1 Simulation through Geometric Brownian Motion

An initial approach to scenario tree generation involves modeling the stock dynamics with the Geometric Brownian Motion; thus, the assumption of this methodology is that the stocks' prices process follows the SDE of Equation 1.2, which, as shown above, features as solution the formula given by:

$$S_t = S_0 \cdot e^{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t}.$$

The key aspects of this approach are outlined in the following steps :

- at each branching step, increments for log-returns of stocks are sampled from a multivariate normal distribution with mean $\hat{\boldsymbol{\mu}}$ and covariance matrix $\hat{\boldsymbol{\Sigma}}$;
- besides, a check to ensure the absence of arbitrage opportunities in the generated stock prices is performed to guarantee a reliable simulation (see subsection 2.2.3 for more details);
- for the computation of nodes' probabilities while preserving the match with the Brownian Motion moments, a Moment Matching problem is solved;
- lastly, options in the hedging portfolio are priced according to the generated underlyings prices through the B&S formulas (Equation 1.4 and Equation 1.5).

Moment matching for probabilities computation

As anticipated, a Moment Matching problem is solved for the computation of scenario tree node probabilities, matching some scenario tree properties with the corresponding properties estimated from historical data.
To better clarify how scenario tree moments are computed, consider the case of Figure 2.2, where a parent node n_0 is followed by c children nodes $\{n_i\}_{i=1\ldots,c}$.

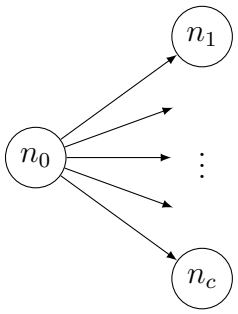


Figure 2.2: Branching step in a scenario tree from a parent node n_0 to his children nodes $\{n_i\}_{i=1,\ldots,c}$.

Given the following:

- the log-return of the j^{th} stock price from parent node n_0 to the child node n_i

$$y_j^{(n_i)} = \ln\left(\frac{S_j^{(n_i)}}{S_j^{(n_0)}}\right),$$

- the vector of log-returns of stock j from n_0 to his children nodes

$$\boldsymbol{y}_j = \left[y_j^{(n_1)}, \ldots, y_j^{(n_c)}\right],$$

- the vector $\boldsymbol{\pi}$, equal to the conditional probability of each children node given the parent node, that is

$$\boldsymbol{\pi} = [\pi_1, \ldots, \pi_c] = [\mathbb{P}(n_1 \mid n_0), \ldots, \mathbb{P}(n_c \mid n_0)],$$

then, for each non-leaf node in the scenario tree a moment matching problem is solved to compute children nodes probabilities, where the matched moments are:

1. the first moment of log-returns in scenario tree

$$\mathbb{E}[\boldsymbol{y_j}] = \boldsymbol{\pi}^T \boldsymbol{y}_j = \sum_{i=1}^{c} \pi_i \ln\left(\frac{S_j^{(n_i)}}{S_j^{(n_0)}}\right)$$

   with the first moment of the BM followed by log-returns

$$M_1(j) = \left(\hat{\mu}_j - \frac{\hat{\sigma}_j^2}{2}\right) dt$$

   for each stock j;

2. the second moment of log-returns in scenario tree

$$\mathbb{E}[\boldsymbol{y}_j^2] = \boldsymbol{\pi}^T \boldsymbol{y}_j^2 = \sum_{i=1}^{c} \pi_i \left[\ln\left(\frac{S_j^{(n_i)}}{S_j^{(n_0)}}\right)\right]^2$$

   with the second moment of the BM of log-returns

$$M_2(j) = \hat{\sigma}_j^2 dt + \left[\left(\hat{\mu}_j - \frac{\hat{\sigma}_j^2}{2}\right) dt\right]^2$$

   for each stock j;

3. and the expectation of the product between log-returns of two different stocks j and k

$$\mathbb{E}[\boldsymbol{y}_j \cdot \boldsymbol{y}_k] = \boldsymbol{\pi}^T(\boldsymbol{y}_j \cdot \boldsymbol{y}_k) = \sum_{i=1}^{c} \pi_i \left[\ln\left(\frac{S_j^{(n_i)}}{S_j^{(n_0)}}\right)\right] \cdot \left[\ln\left(\frac{S_l^{(n_i)}}{S_l^{(n_0)}}\right)\right]$$

with the correspondent property of the BM

$$P_1(j,k) = (\hat{\sigma}_j \hat{\sigma}_k dt) \cdot \hat{\rho}_{j,k} + \left(\hat{\mu}_j - \frac{\hat{\sigma}_j^2}{2}\right) dt \cdot \left(\hat{\mu}_k - \frac{\hat{\sigma}_k^2}{2}\right) dt$$

for each pair of stocks i, j.

The optimization problem solved for the computation of nodes probabilities is:

$$\begin{aligned} \min_{\boldsymbol{\pi}} \quad & \sum_{j=1}^{n} \left[(M_1(j) - \mathbb{E}[\boldsymbol{y}_j])^2 + \left(M_2(j) - \mathbb{E}[\boldsymbol{y}_j^2]\right)^2 + \sum_{k=j+1}^{n} \left(P_1(j,k) - \mathbb{E}[\boldsymbol{y}_j \cdot \boldsymbol{y}_k]\right)^2 \right] \\ \text{s.t.} \quad & \sum_i \pi_i = 1, \\ & \pi_i \in [l, u] \quad \forall\, i, \end{aligned}$$

where $0 < l \ll u < 1$ represent bounds on probabilities values (needed for numerical stability in the implementation). Since the problem is formulated as a constrained Quadratic Program (QP), it is solved using the solver Gurobi[2].

Alternative formulation. Also an alternative formulation of the moment matching problem is implemented for the probability computation, where the difference lies in the third moment matched: indeed, the third property is replaced by the correlation matrix $\boldsymbol{\rho}$, matched with the historical correlation $\hat{\boldsymbol{\rho}}$. The pairwise correlation between log-returns of stocks j and k in the scenario tree is computed as:

$$\rho_{j,k} = \frac{\boldsymbol{\pi}^T \left[(\boldsymbol{y}_j - \mu_j) \cdot (\boldsymbol{y}_k - \mu_k)\right]}{\sigma_j \cdot \sigma_k},$$

where μ_j is the -already introduced- term $\mathbb{E}[\boldsymbol{y}_j]$, while

$$\sigma_j = \sqrt{\boldsymbol{\pi}^T \left(\boldsymbol{y}_j - \mathbb{E}[\boldsymbol{y}_j]\right)^2}$$

(the same hold for μ_k and σ_k). The objective of this formulation is:

$$\min_{\boldsymbol{\pi}} \quad \|\mathbb{E}[\boldsymbol{y}] - M_1\|_2 + \|\mathbb{E}[\boldsymbol{y}^2] - M_2\|_2 + \|\boldsymbol{\rho} - \hat{\boldsymbol{\rho}}\|_1,$$

[2]Gurobi is an optimization solver used to solve linear, integer and quadratic programming problems under given constraints.

with the same set of constraints of the previous formulation. The inclusion of the correlation in the objective function leads to an optimization model that is not a LP nor a QP, thus not resolvable with Gurobi solver: for this implementation of Moment Matching, the Sequential Least Squares Programming (SLSQP) solver is adopted.

2.2.2 Simulation through Moment Matching

The second approach implemented for scenario tree generation involves the solution of a Moment Matching problem for the computation of both node probabilities and stock prices. Recalling the structure given by Figure 2.2, in this context decision variables are represented by both $\boldsymbol{\pi}$ and $\boldsymbol{y}_j$ (for each stock j) and the matched moments are:

1. $\mu_j = \mathbb{E}[\boldsymbol{y_j}]$ with the first moment of log-returns $\hat{\mu}_j$ estimated from historical data;
2. the standard deviation of log-returns in scenario tree
$$\sigma_j = \sqrt{\boldsymbol{\pi}^T(\boldsymbol{y}_j - \mu_j)^2}$$
with the standard deviation of log-returns $\hat{\sigma}_j$ estimated from historical data;
3. the correlation among each pair of stocks
$$\rho_{j,k} = \frac{\boldsymbol{\pi}^T\left[(\boldsymbol{y}_j - \mu_j)\cdot(\boldsymbol{y}_k - \mu_k)\right]}{\sigma_j \cdot \sigma_k}$$
and the historical correlation $\hat{\rho}_{j,k}$;
4. the skewness
$$\gamma_j = \boldsymbol{\pi}^T\left(\frac{\boldsymbol{y}_j - \mu_j}{\sigma_j}\right)^3$$
and the historical skewness $\hat{\gamma_j}$;
5. and the kurtosis
$$\nu_j = \boldsymbol{\pi}^T\left(\frac{\boldsymbol{y}_j - \mu_j}{\sigma_j}\right)^4$$
and the historical kurtosis $\hat{\nu}_j$.

The optimization model ends up to be:

$$\begin{aligned}
\min_{\boldsymbol{\pi},\boldsymbol{y}} \quad & \|\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}\|_2 + \|\boldsymbol{\sigma} - \hat{\boldsymbol{\sigma}}\|_2 + \|\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}}\|_2 + \|\boldsymbol{\nu} - \hat{\boldsymbol{\nu}}\|_2 + \|\boldsymbol{\rho} - \hat{\boldsymbol{\rho}}\|_1 \\
\text{s.t.} \quad & \sum_i \pi_i = 1, \\
& \pi_i \in [l, u] \quad \forall\ i,
\end{aligned}$$

Due to its nature, the problem is solved using Sequential Least Squares Quadratic Programming (SLSQP). In the same way as for the simulation with GBM, after the scenario tree generation, a check to ensure the absence of arbitrage opportunities is performed and finally options are priced.

This approach is quite flexible as it does not assume any particular underlying dynamics for the stocks' prices: the implication is the possibility to include skewness and kurtosis in the matched moments (while for the GBM simulation they could not be matched with the historical ones as for the Gaussian assumption skewness and kurtosis are fixed in the generated values of the scenario tree).

2.2.3 Arbitrage-free scenarios

The generation of a scenario tree in financial applications is as useful as delicate: during the generation of assets' prices it must be guaranteed the condition on the absence of arbitrage opportunities. Without entering into details, note that the absence of arbitrage opportunities in our application can be achieved by checking their absence in simulated stock prices, disregarding option prices (which are determined according to the stock prices).

An *arbitrage opportunity* is a strategy which does not require any initial investments and allows to generate, in at least one future state, a positive cash flow, with a non-negative cash flow in every other future state. Formally, it can be seen as a portfolio h with value V_t^h at time t for which it holds that, for a set Ω of possible future scenarios at time $t+1$:

$$\begin{aligned} &V_t^h = 0, \\ &V_{t+1}^h(\omega) \geq 0 \quad \forall \omega \in \Omega, \\ &\mathbb{E}_\Omega[V_{t+1}] > 0. \end{aligned}$$

Before analyzing how arbitrage opportunities can be avoided in scenario tree generation, the definition of a *dominant strategy* should be introduced. The latter is a strategy with a stronger condition with respect to the previous one: indeed, a dominant strategy can be viewed as a portfolio h for which it holds that

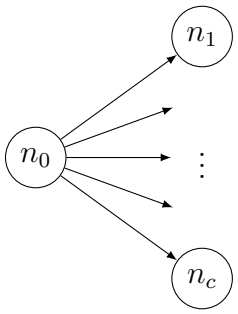
$$\begin{aligned} &V_t^h = 0, \\ &V_{t+1}^h(\omega) > 0 \quad \forall \omega \in \Omega, \end{aligned}$$

or, if there is a risk-free asset in the market,

$$V_t^h < 0, \tag{2.2}$$

$$V_{t+1}^h(\omega) \geq 0 \quad \forall \omega \in \Omega, \tag{2.3}$$

Let's discuss first how to ensure the absence of dominant strategies in the generated scenario tree. Recall again the case of Figure 2.2:



Consider the following:

- the vector of cash and stocks prices at the parent node

$$\boldsymbol{p}^{(n_0)} = [p_0^{(n_0)}, \ldots, p_m^{(n_0)}];$$

- the vector of cash and stocks prices at the child node n_i

$$\boldsymbol{p}^{(n_i)} = [p_0^{(n_i)}, \ldots, p_m^{(n_i)}];$$

- a portfolio $\boldsymbol{h} = [h_0, \ldots, h_m]$, containing the holdings of the risk-free rate and the m stocks, whose value at each node in the set $\{n_i\}_{i=0,\ldots,c}$ is given by

$$V_{n_i}^h = \boldsymbol{h}^T \boldsymbol{p}^{(n_i)}.$$

The check on the absence of dominant strategies, according to Equation 2.2 and Equation 2.3, is accomplished by studying the solution of the following LP problem:

$$\begin{aligned} \min_{\boldsymbol{h}} \quad & \boldsymbol{h}^T \boldsymbol{p}^{(n_0)} \\ \text{s.t.} \quad & \boldsymbol{h}^T \boldsymbol{p}^{(n_i)} \geq 0, \quad \forall\, i = 1, \ldots, c \end{aligned}$$

The dual of this problem is:

$$\max_{\boldsymbol{\lambda}} \quad 0 \tag{2.4}$$

$$\text{s.t.} \quad \boldsymbol{\lambda}^T \mathbf{p}_j = p_j^{(n_0)}, \quad \forall \text{ asset } j \tag{2.5}$$

$$\boldsymbol{\lambda} \geq 0 \tag{2.6}$$

where $\mathbf{p}_j = [p_j^{(n_1)}, ..., p_j^{(n_c)}]$ indicates the vector of prices of asset j at children nodes and $\boldsymbol{\lambda}$ is the vector of dual variables.

Note that the dual is a *feasibility problem*, meaning that the objective function is a constant value: the actual purpose of this problem is to verify the existence of a solution $\boldsymbol{\lambda}^*$ for which the condition of the constraints is met, disregarding the minimization of an objective value. If such a solution exists, then the value at the optimum of the objective function is zero both for the dual and the primal problem, indicating that the generated prices for the children nodes allow for no dominant strategy; additionally, it can be proven that if a stronger condition is met, that is the solution $\boldsymbol{\lambda}^*$ is strictly positive, then not only there are no dominant strategies, but also arbitrage opportunities are avoided.
It can also be proven that if the problem has a feasible solution, then there exists a probability measure $\boldsymbol{q}$ (obtained rescaling $\boldsymbol{\lambda}^*$) under which the expected value of the discounted price process is constant, that is:

$$\mathbb{E}_{\boldsymbol{q}}\left[\frac{\boldsymbol{p}_j}{e^{r\Delta t}}\right] = \frac{\boldsymbol{q}^T\boldsymbol{p}_j}{e^{r\Delta t}} = p_j^{(n_0)}. \tag{2.7}$$

The reason behind these results lies in the concept of a martingale.

Martingale measure.

The concept of martingale is recurrent in financial literature since it is related to the risk-neutral and arbitrage-free pricing of financial derivatives.
Without loss of generality, a stochastic process M_n is said to be a **martingale** if

$$\mathbb{E}[M_{n+1}|\mathcal{F}_n] = M_n. \tag{2.8}$$

Here, the filtration $\mathcal{F}_n$ is formally defined as:

$$\mathcal{F}_n = \sigma(M_0, ..., M_n),$$

that is, the sigma-algebra generated by the process realizations up to time n, meaning that it is the set containing all the events of the stochastic process observed up to time n. Briefly, $\mathcal{F}_n$ represents the overall information about the process history available up to time n. Equation 2.8 states that, conditioned on filtration $\mathcal{F}_n$, the expected future value of the process M_{n+1} is equal to the current value M_n.

With the previous definition provided, let us now define a martingale measure. Given a stochastic process X_n with a probability measure $\mathbb{P}$ (known as physical or real-world measure), the **equivalent martingale measure**, also denoted as risk-neutral measure, is a probability measure $\mathbb{Q}$ under which the stochastic process X_n is a martingale. To better clarify, consider that for any stochastic processes Equation 2.8 may not be satisfied, while there might exists a probability measure $\mathbb{Q}$ under which it is satisfied. In formulae:

$$\mathbb{E}_{\mathbb{P}}[X_{n+1}|\mathcal{F}_n] \neq X_n, \quad \mathbb{E}_{\mathbb{Q}}[X_{n+1}|\mathcal{F}_n] = X_n.$$

Returning back to our framework, it can be proven that the absence of arbitrage in financial markets is strictly related to the existence of (at least[3]) one martingale measure for the discounted price process. That is, denoting by $P_{t,j}$ the price process of an asset j, we are wondering whether there exists a probability measure $\mathbb{Q}_t$ under which

$$\mathbb{E}_{\mathbb{Q}_t}\left[\frac{P_{t+1,j}}{e^{r\Delta t}} \;\middle|\; \mathcal{F}_t\right] = P_{t,j}.$$

This condition translates into Equation 2.7 related to the optimization problem presented above and the existence of a solution $\boldsymbol{\pi}^*$ to the problem corresponds to the existence of a martingale measure which ensures the absence of arbitrage in simulated scenarios of the tree.

Alternative approach to arbitrage-free scenarios.

M.Villaverde shows in [1] a different method to ensure absence of arbitrage in simulated scenarios; in particular, in his methodology, the moment matching method with a condition of absence of arbitrage are combined in the generation of tree's values. Based on his approach, the following problem is taken in consideration. Recalling the structure with a parent node n_0 and c children nodes $\{n_1, \ldots, n_c\}$, after an initial generation of stocks' prices $\{\bar{S}_j^{(n_i)}\}$, for each stock j the generated values are modified according to the solution of:

$$\min_{\{S_j^{(n_i)}\}} \quad \sum_i (S_j^{(n_i)} - \bar{S}_j^{(n_i)})^2 \tag{2.9}$$

$$\text{s.t.} \quad \frac{S_j^{\max} - S_j^{(n_0)}}{S_j^{(n_0)}} \geq \alpha \tag{2.10}$$

$$\frac{S_j^{\min} - S_j^{(n_0)}}{S_j^{(n_0)}} \leq -\alpha \tag{2.11}$$

where α is a small positive number and the two terms $S_j^{\max}$ and $S_j^{\min}$ correspond to the decision variables of the stock price respectively in the child node n_i where the generated stock price $\bar{S}_j^{(n_i)}$ was the maximum and minimum. This optimization problem should guarantee that at least one stock price among the children nodes increased with respect to the previous price $S_j^{(n_0)}$ and, viceversa, at least one stock price decreased.

[3]If such a martingale measure not only exists but it is also unique, the market is not only arbitrage-free but also complete, meaning that there exists a replicating strategy for each derivative in the market.

Both strategies have been implemented in the project and tested: both resulted in effective results in terms of performance, allowing to generate a scenario tree in absence of arbitrage; moreover, despite the fact that the first method requires a new generation of prices for children nodes for each time an arbitrage opportunity is detected (resulting in a loop of values generation until an arbitrage-free scenario is found), both method performed well in terms of computational time, with the first approach faster in some cases and, viceversa, slower in other cases.

Chapter 3

Optimization Model

The previous chapters provided some key concepts useful to understand the hedging strategy proposed in this thesis. The first chapter introduced the topic and established the financial market setup in which the strategy is implemented, detailing the financial instruments and their assumed dynamics. The second chapter, instead, presented the fundamentals of Stochastic Optimization and scenario tree generation, which are leveraged to model the uncertainty about possible future outcomes and are used to solve the hedging problem at each decision stage.

In this third chapter, the hedging strategy is formally presented. The mathematical formulation of the optimization model solved at each decision stage of the dynamic hedging process is discussed, specifying the decision variables, constraints and the objective function involved. Following this, various alternative configurations of the hedging problem are introduced, each corresponding to a specific choice for the strategy to implement. A discussion on different approaches to portfolio financing will be provided, along with an analysis of how different levels of risk aversion can be modeled. Lastly, the super-replication hedging problem is introduced as it will be leveraged in later chapters to benchmark the performance of the proposed strategy under higher levels of risk aversion.

Before analyzing the optimization problem, the Asset-Liability Management (ALM) problem is introduced, as it provides a framework for structured approaches for the management of a portfolio while ensuring that future liabilities are met: this represents the general framework within which the hedging problem is situated.

3.1 Conceptual Framework

As mentioned before, shorting a position in the target asset implies incurring a future liability if the option is in-the-money at maturity; to hedge against the potential liability, the premium received from selling the option contract can be used to create a hedging portfolio which aims at replicating the option's payoff; in this way, the liability can be balanced by the cash flow generated by the hedging portfolio. In this framework, a multi-stage stochastic programming based asset-liability management (ALM) optimization model is proposed to address the hedging problem.

At the initial time, a short position in the target asset is assumed: the option premium gained is a cash flow available to be reinvested in the construction of a hedging portfolio. The strategy is to rebalance the portfolio a finite number of times before the option's maturity, allowing the hedge to be calibrated according to the market and assets dynamics. At each rebalancing time, the portfolio is optimized according to an optimization problem presented in what follows (subsection 3.2.1). To model the uncertainty in the market evolution, a scenario tree is simulated in order to have a perspective of what the future possible outcomes of the financial instruments are. According to the simulation, the model is solved trying to reach a hedging portfolio whose final value does not deviate much from the target asset payoff.

A relevant aspect to be considered is the presence of transaction costs (market frictions known to work against hedging strategies) included in the model to better reflect real financial market conditions.

3.1.1 The Asset-Liability Management problem

An Asset-Liability Management (ALM) problem is a financial optimization model which aims at optimizing the allocation of a wealth over a portfolio of different assets in view of meeting future liabilities over time, subject to various constraints. Liabilities can be seen as expected cash outflows along the time horizon, while assets in the hold portfolio may be seen as expected cash inflows: the objective is to balance the two, minimizing the risk exposure associated with liabilities and potential losses that may arise from them.
As expressed by Y.Romanyuk in [3], strategies for asset-liability management may be modeled as single-period or multi-period problems and may be static rather than stochastic. Concerning multi-period stochastic strategies, which is the case of our hedging problem, these models allow both assets and liabilities to evolve stochastically over time; investors, thus, should construct a portfolio at the initial

stage and rebalance its composition along the time horizon in determined time instants according to the evolution of assets and liabilities. The general idea of an ALM model is represented graphically in Figure 3.1.

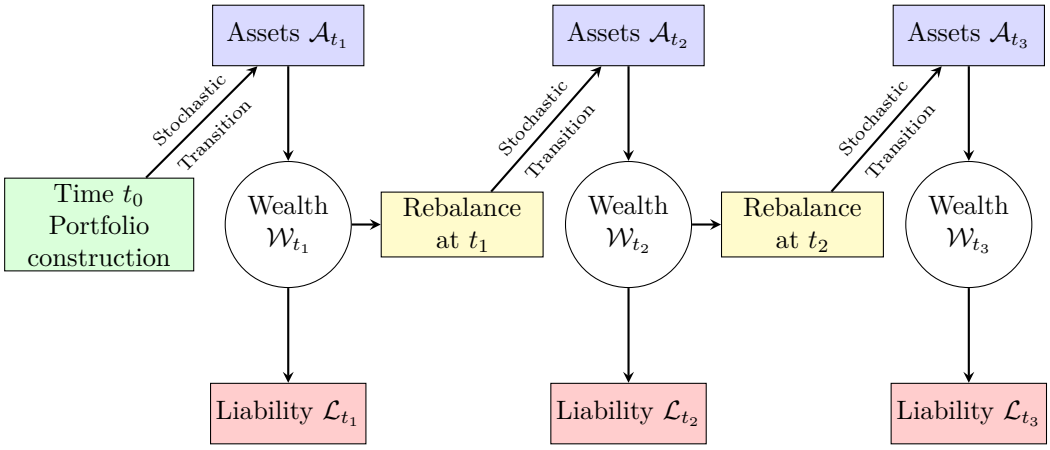


Figure 3.1: Flowchart for a three-stages stochastic asset-liability management problem.

3.2 Mathematical formulation of the hedging problem

As introduced, the hedging problem is formulated as an asset-liability management problem lying within the multi-stage stochastic optimization framework. The following key points should be kept in mind:

- at maturity date, a liability may occur if the holder of the target asset exercises it;
- after an initial portfolio construction at decision stage 0, portfolio rebalancing occur at each stage up to the time-step before maturity: at maturity date, the hedging portfolio is used to offset the potential liability without any further rebalancing;
- possibly, intermediate cash outflows may occur before maturity due to intermediate liabilities (think for example to coupons) and the optimization model should be capable to take them into account;
- rebalancing operations occur with transaction costs proportional to the traded volume of assets;

In what follows, the optimization problem and its characteristics are introduced.

3.2.1 Definition of variables and parameters

Let the following indices represent the problem features:

- $j \in \{1, ..., n\}$ be the index for hedging assets;
- $i \in \{0\} \cup \mathcal{N} \cup \mathcal{T}$ be the index for nodes in the scenario tree, where:
  - $i = 0$ corresponds to the root node;
  - $\mathcal{N}$ is the set of non-leaf nodes, excluding the root node;
  - $\mathcal{T}$ is the set of terminal nodes, i.e. leaf nodes.

Let's define the following parameters:

- $\boldsymbol{p}_j^{(i)}$: price of asset j at node i.
- $\Phi_T^{(i)}$: payoff of the target asset at the terminal node $i \in \mathcal{T}$.
- $\boldsymbol{\pi}_T^{(i)}$: probability of the terminal node $i \in \mathcal{T}$, that is the probability of the path from the root node to the leaf node i.
- W_0: initial wealth available for the construction of the hedging portfolio;
- $\mathbf{c}_j$: transaction cost of asset j;
- $\mathrm{a}(i)$: index referring to the parent node of node $i \in \mathcal{N} \cup \mathcal{T}$;
- $\boldsymbol{q}_j$: previous quantity of asset j, taken to be the the quantity of holdings of asset j held at the current stage before solving the problem. Note that for the model solved at stage 0 (when the portfolio is still to be constructed) $\boldsymbol{q}_j = 0 \, \forall j$, while for problems solved at further stages $\boldsymbol{q}_j$ is equal to the holdings of asset j found by solving the model at the previous stage;

and decision variables:

- $\mathbf{x}_j^{(i)}$: quantity of asset j bought at node $i \in \mathcal{N} \cup \{0\}$.
- $\mathbf{y}_j^{(i)}$: quantity of asset j sold at node $i \in \mathcal{N} \cup \{0\}$.
- $\mathbf{z}_j^{(i)}$: holdings of asset j at node $i \in \mathcal{N} \cup \{0\}$.
- $\mathbf{e}_i^+$, $\mathbf{e}_i^-$: positive and negative error terms at leaf node $i \in \mathcal{T}$ of the scenario tree.

3.2.2 Optimization model

The hedging optimization problem solved at decision stage 0 is formulated as follow [1]:

$$\min \quad \sum_{i\in\mathcal{T}} \left(\mathbf{e}_i^+ + \mathbf{e}_i^-\right) \cdot \boldsymbol{\pi}_T^{(i)} \tag{3.1}$$

$$\text{s.t.} \quad \mathbf{x}_j^{(0)} - \mathbf{y}_j^{(0)} + \boldsymbol{q}_j = \mathbf{z}_j^{(0)}, \quad \forall j \tag{3.2}$$

$$\mathbf{z}_j^{(\mathrm{a}(i))} + \mathbf{x}_j^{(i)} - \mathbf{y}_j^{(i)} = \mathbf{z}_j^{(i)}, \quad \forall i \in \mathcal{N},\ \forall j \tag{3.3}$$

$$W_0 + \sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{y}_j^{(0)} - \sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{x}_j^{(0)} = 0, \tag{3.4}$$

$$\sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(i)}\cdot\mathbf{y}_j^{(i)} - \sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(i)}\cdot\mathbf{x}_j^{(i)} = 0, \quad \forall i \in \mathcal{N} \tag{3.5}$$

$$\mathbf{e}_i^+ - \mathbf{e}_i^- = \sum_j \mathbf{p}_j^{(i)}\cdot\mathbf{z}_j^{(\mathrm{a}(i))} - \Phi_T^{(i)}, \quad \forall i \in \mathcal{T}. \tag{3.6}$$

$$\mathbf{x}_j^{(i)}, \mathbf{y}_j^{(i)} \geq 0, \quad \forall i \in \mathcal{N}\cup\{0\},\ \forall j \tag{3.7}$$

$$\mathbf{e}_i^+, \mathbf{e}_i^- \geq 0, \quad \forall i \in \mathcal{T} \tag{3.8}$$

For problems solved at subsequent decision stages, only the constraint given by Equation 3.4 changes and its formulation will depend on whether the strategy is defined as self-financing or not: further details will be provided in subsection 3.3.1.

Let us look into details each component of the problem.

Objective Function.

Recalling the definition of the hedging error as the difference, at maturity, between the target asset payoff and the value of the hedging portfolio, the goal of the objective function (3.1) is to minimize the expected hedging error over the set of simulated scenarios determined by the leaf nodes $\mathcal{T}$ of the scenario tree.
The hedging error is decomposed into its positive and negative parts $\mathbf{e}_i^+$ and $\mathbf{e}_i^-$: this decomposition allows for asymmetrical penalization of positive and negative hedging errors: see subsection 3.3.2 for additional details.

Inventory Balance Constraints.

For each asset, constraints (3.2) and (3.3) ensure that holdings at each node are consistent with the quantities bought, sold and already held:

- for the root node, given the equation

$$\mathbf{x}_j^{(0)} - \mathbf{y}_j^{(0)} + \boldsymbol{q}_j = \mathbf{z}_j^{(0)} \quad \forall j,$$

it holds that, for each asset j, the quantity bought minus the quantity sold, plus the holding already held, is equal to the asset holding of the root node. As mentioned above, it holds trivially that $\boldsymbol{q}_j = 0 \ \forall j$ when solving the model at stage 0;

- for all other non-leaf nodes $i \in \mathcal{N}$ it holds the equation

$$\mathbf{z}_j^{(\mathrm{a}(i))} + \mathbf{x}_j^{(i)} - \mathbf{y}_j^{(i)} = \mathbf{z}_j^{(i)} \quad \forall i \in \mathcal{N}, \ \forall j,$$

  according to which for each asset j the holding inherited from the parent node plus the ones bought minus the ones sold is equal to the holding of the current node.

At terminal nodes $i \in \mathcal{T}$ no portfolio rebalancing is allowed, thus for each leaf node the holdings correspond to those of the parent node.

Cash balance constraints.

The model exhibits two cash balance constraints, specifically constraints (3.4) and (3.5). The former, according to which

$$W_0 + \sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)} - \sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} = 0,$$

imposes that, for the root node of the tree, the total cost of purchase minus the earnings from the sales should be equal to the initial wealth W_0; this means that the portfolio should be constructed by buying and selling asset holdings incurring in a total cost that is equal to the given wealth. Different versions of Constraint (**??** will be provided in subsection 3.3.3, which discusses about the initial wealth W_0.
On the other hand, the second constraint, whose equation is

$$\sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(i)} \cdot \mathbf{y}_j^{(i)} - \sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(i)} \cdot \mathbf{x}_j^{(i)} = 0, \quad \forall i \in \mathcal{N}$$

requires the portfolio to be self-financing at each non-leaf node (excluded the root node), allowing for no cash flows from or to the outside: any changes in the portfolio value must be financed through adjustments within the portfolio itself. All cash flows listed in these constraints account for transaction costs c_j (one for each asset j); note that when selling holdings the transaction cost is applied with the multiplicative factor $(1 - c_j)$, while when buying the factor $(1 + c_j)$ is used.

Possibly, intermediate liabilities $\mathcal{L}_t$ during the hedging horizon may occur: these are

trivially managed by the optimization problem by adding a negative cash flow in cash balance contstraints; indeed, constraints (3.4) and (3.5) would be respectively restated as:

$$\sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{y}_j^{(0)}-\sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{x}_j^{(0)}-\mathcal{L}_0\leq 0$$

and

$$\sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(i)}\cdot\mathbf{y}_j^{(i)}-\sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(i)}\cdot\mathbf{x}_j^{(i)}-\mathcal{L}_{t(i)}=0,\quad\forall i\in\mathcal{N},$$

where $t(i)$ refers to the time-step corresponding to node i of the scenario tree.

Error definition constraint.

The error definition is a key element in the optimization model as it helps shaping the objective function to be minimized. The constraint given by Equation 3.6, restated below,

$$\mathbf{e}_i^+-\mathbf{e}_i^-=\sum_j\mathbf{p}_j^{(i)}\cdot\mathbf{z}_j^{(\mathrm{a}(i))}-\Phi_T^{(i)},\quad\forall i\in\mathcal{T},$$

defines the hedging error at each leaf node of the scenario tree as the difference between the total portfolio value and the payoff of the target asset, whose position is aimed to be hedged. The resulting error is then decomposed into its positive and negative parts ($\mathbf{e}^+$ and $\mathbf{e}^-$); the reason behind this approach is to discriminate between two types of errors:

- if the portfolio value falls short of the target payoff, i.e.

  $$\sum_j\mathbf{p}_j^{(i)}\cdot\mathbf{z}_j^{(\mathrm{a}(i))}<\Phi_T^{(i)},$$

  then $\boldsymbol{e}_i^+=0$ and the total error $\boldsymbol{e}_i$ at terminal node i is negative and equal in absolute value to $\boldsymbol{e}_i^-$. This error is considered a shortfall, meaning that the payoff replication has not been met and a loss has incurred in our hedging strategy;

- on the other hand, if the portfolio value exceeds the target payoff, that is the case when

  $$\sum_j\mathbf{p}_j^{(i)}\cdot\mathbf{z}_j^{(\mathrm{a}(i))}>\Phi_T^{(i)},$$

  then $\boldsymbol{e}_i^-=0$ and the total error e_i at leaf node i is positive and equal to $\boldsymbol{e}_i^+$. Keeping in mind the purpose of the hedging problem (which involves

requiring a payoff-replication condition), this error is considered a surplus, meaning that the portfolio not only meets the target payoff but also exceeds it. One may argue that this situation should not be penalized, as the payoff is replicated and the portfolio ends up with a surplus value; however, it is important to note that constructing a portfolio with a higher value requires a greater investment, which leads to a higher cost of the hedging strategy.

This formulation allows the hedger to track the performance of the strategy by analyzing both surplus and shortfall deviations of the portfolio value with respect to the target asset's payoff, while also allowing for the introduction of asymmetric penalties in the objective function to better reflect the hedger's risk aversion (as will be discussed later in subsection 3.3.2).

3.3 Alternative formulations

Given the primal model formulation described above, some alternative formulations are introduced to account for various aspects of the hedging strategy according to the hedger's preferences.

3.3.1 Financing strategies

Self-financing vs non-self-financing strategy

Given the following notation, according to which $V_t^h(\tau)$ denotes the value at time t of the held portfolio h constructed at time $\tau \leq t$, a self-financing strategy is a portfolio rebalancing strategy according to which:

$$V_t^h(t - \Delta t) = V_t^h(t).$$

In practice, the latter is a budget equation imposing that changes in the portfolio composition between time instants $t - \Delta t$ and t should be completely financed by the reallocation of existing wealth among the assets in the portfolio rather than by financing the portfolio from the outside; thus, the rebalancing operation requires no exogenous cash flow.

The optimization model at decision stages following the stage 0 can be formulated with both a self-financing portfolio and a non-self-financing one, depending on the hedger preference.
In the self-financing case, except for the problem at stage 0 where the portfolio needs an initial investment to be constructed, constraint (3.4) should be substituted with:

$$\sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)} - \sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} == 0,$$

while in the non-self-financing case it should be substituted with:

$$\sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{y}_j^{(0)}-\sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{x}_j^{(0)}<=0.$$

Note that for the non-self-financing case, it is allowed only additional financing of the portfolio: the possibility to withdraw part of the wealth previously invested in the hedging portfolio is discussed in a following section.

To better understand this concept, some considerations are now made. The distinction between self-financing and non-self-financing strategies is found in how the wealth is managed along the hedging horizon. As anticipated, a self-financing strategy requires that any changes in the portfolio's composition is financed entirely by the reallocation of existing wealth; thus, after the initial investment in the portfolio construction, assets holdings are bought or sold without the need for additional exogenous capital but only by redistributing the portfolio's current wealth. Mathematically, this constraint implies that negative cash flows due to purchasing new assets are balanced by the positive flows from selling existing ones. On the other hand, a non-self-financing strategy allows for external cash inflows, meaning that the hedger can inject additional capital to adjust the portfolio composition. This flexibility may be advantageous in scenarios where market conditions demand adjustments of the portfolio composition that cannot be financed entirely by existing wealth. In light of this, in the applications it is expected that the non-self-financing case will exhibit better performance in terms of hedging, as the hedger can finance the portfolio at subsequent stages in the dynamic programming process, leading to a more optimal strategy, hopefully reducing hedging errors and enhancing the effectiveness of the risk management. However, it cannot be excluded that the possibility of further financing the portfolio at intermediate stages may introduce additional variance in the total loss.

Withdrawal possibility

One aspect to take into account before solving the model is the possibility to withdraw money from the strategy along its lifetime. In practice, we are talking about the possibility to rebalance the portfolio in such a way that a cash flow is generated from the strategy to the outside, hence returning part of the wealth originally invested in the portfolio construction to the hedger.
The withdrawal of wealth translates into the inequality:

$$\sum_{j=1}^{n}(1-\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{y}_j^{(0)}-\sum_{j=1}^{n}(1+\mathbf{c}_j)\cdot\mathbf{p}_j^{(0)}\cdot\mathbf{x}_j^{(0)}\geq 0,$$

that is, the cash balance of the rebalancing operation generates a cash flow which is against the strategy but in favor of the hedger, who would receive back part of

the initial capital invested. Indeed, the amount gained by the assets' sale is set greater than the amount invested in the current rebalancing operation, resulting in a reduction of the portfolio value equal to the cash flow that returns to the hedger.

Some skepticism may arise around the withdrawal concept; in particular, let us consider two aspects.

Why should money be withdrawn after an initial investment? Let's make an example: from the perspective of a bank, the latter may manage simultaneously many different financial strategies, thus a wealth allocation among these strategies should be optimized. If the hedging strategy of the target asset is proceeding well (meaning that the initial wealth invested may exceed what is needed along the hedging horizon for the rebalancing operations), the bank can decide to re-allocate part of its wealth, withdrawing wealth from the hedging strategy and invest it in another financial operation, be it currently active or still to be executed. Thus, in this context, withdrawing wealth from the strategy may turn out to be advantageous.
The same holds for any hedger who might use the model presented in this work.

How could the possibility of withdrawal enhance the performance? To answer this question, it is useful to recall the objective of our model. The latter aims at hedging against a short position in the target asset through a (constrained) replication problem, where the object of replication is the target asset payoff (in truth, it is not a strict replication, but rather a relaxed one, allowing for the presence of a replication error term). Considering this, let us analyze the behavior of the hedging strategy in a scenario where: at the beginning of the hedging horizon, the target asset is in-the-money, while market deviations lead it to be at-the-money/out-of-money when reaching the maturity date. This situation is critical for the following reason: at stage 0 the hedging portfolio is constructed with an initial investment according to the solution found by the stochastic optimization model. Without withdraw possibility, as the hedging horizon proceeds and the target asset loses value, the model will force the portfolio to track the target asset; thus, the portfolio itself will lose part of its value[1]; this is motivated by the fact that the model's objective function penalizes not only shortfall values but also surplus values. As a result, giving the model the possibility of withdrawal, situations like the one mentioned do not let the portfolio to lose value, but rather it would allow

[1]This could be achieved by finding a portfolio rebalancing strategy focused on losing value, thus betting on a portfolio composition which is not promising according to the scenario tree generated.

the hedger to withdraw the quantity of money in excess and invest it at a risk-free rate of interest or in another financial strategy.

3.3.2 Risk-aversion parameter

In order to better reflect the risk aversion of the hedger, a risk aversion parameter $\gamma \in [0,1]$ is introduced in the objective function trading-off the weight of the negative and positive part of the hedging error to minimize. The objective function for this alternative formulation of the model becomes:

$$\sum_{i \in \mathcal{T}} \left(\gamma \mathbf{e}_i^+ + \mathbf{e}_i^- \right) \cdot \boldsymbol{\pi}_T^{(i)}.$$

Recalling that the error decomposition allows the introduction of asymmetry in the objective function penalization, the presence of γ reflects this asymmetry allowing the model to trade-off between positive and negative hedging errors, reflecting different risk-aversion levels: a lower γ leads the model to focus on minimizing hedging shortfalls, while a higher γ leads for a more balanced penalization of the two parts; indeed:

- with $\gamma = 0$ the objective function boils down to:

  $$\sum_{i \in \mathcal{T}} \mathbf{e}_i^- \cdot \boldsymbol{\pi}_T^{(i)}.$$

  This value of the parameter reflects a more conservative behavior: the whole optimization problem focuses on meeting the target asset's payoff, not minding if there is a surplus in the value of the hedging portfolio; the only focus of our objective lies in the minimization of the expected negative error: this case is the closest to the idea of super-replication (although being different from it), which will be analyzed in section 3.4;

- $\gamma = 1$, instead, reflects a more risk-neutral behavior: those who choose this value for the parameter aim at minimizing the hedging error with no distinction between surplus and shortfall values; what is expected is a solution with hedging errors distributed (nearly) symmetrically around the value 0, with the presence of both under-hedging and over-hedging scenarios;

- intermediate preferences are reflected by the choices of γ between 0 and 1, that is the case when positive and negative hedging errors are penalized asymmetrically, but with a moderate risk aversion.

3.3.3 Initial wealth W_0

An important aspect to take into account when dealing with the problem solved at decision stage 0 is the management of the initial wealth invested in the hedging strategy. As expressed by constraint (3.4), which is reported here for clarity

$$W_0 + \sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)} - \sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} == 0,$$

there is a strict constraint on the value of the initial portfolio investment, meaning that the available wealth W_0 is completely spent in the portfolio construction. However, in some contexts, a relaxed version of this constraint might be preferred, allowing to spend up to W_0, but not strictly requiring it to be fully spent, that is:

$$\sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} - \sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)} \leq W_0.$$

A further generalization of this constraint is when the difference between the amount gained from the sales and the one spent is required to be negative, meaning that there should be an initial investment in the strategy, but without imposing a constraint on its value. This reflects in the constraint:

$$\sum_{j=1}^{n}(1 - \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)} - \sum_{j=1}^{n}(1 + \mathbf{c}_j) \cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} \leq 0. \tag{3.9}$$

In this case, the solver finds a solution in terms of assets' holdings to buy or sell to compose the hedging portfolio and only at this point there is a hint of the actual amount of investment required, given indirectly by the solution. It is important to note that, if Equation 3.9 is used as cash balance constraint in the root node of the problem at stage 0, there would be no restriction on the value of the initial wealth W_0 invested in the portfolio. This may lead to the conclusion that the model can produce a solution that is not affordable in terms of the required investment. However, this approach is based on the assumption that the model can determine in autonomy an optimal investment for the hedging strategy, trading-off the hedging performance and costs related to portfolio construction, rebalancing and transactions[2]. An additional consideration can be made around the expected initial investment: recalling that, in order to hedge against the short position in the target asset, a target-payoff replication model is build and it is expected to face an initial investment that is quite aligned with the price of the target to be hedged; however, further analyses of the results are required to confirm whether the initial wealth selected by the model is indeed close to the actual price of the target asset or if the two deviate.

[2]The hypothesis of an effective trade-off between costs and performance is motivated by the penalization of the surplus term $\boldsymbol{e}_i^+$ in the objective function.

3.4 Super-replication hedging problem

In addition to the optimization model discussed in the previous section, an alternative formulation based on the concept of super-replication is now introduced. The latter is indicated for hedgers with a high level of risk aversion, willing to reach a full hedge over the whole set of simulated scenarios (characteristic that makes the super-replication more conservative and robust also against worst scenarios). The model is presented since it can be useful to compare the behavior of a super-replication model with that of the primal optimization model discussed when the risk aversion parameter γ approaches 0.

The super-replication concept

The idea behind a super-replication problem is to guarantee a complete hedging in every possible future scenario; in mathematical terms, for a discrete set of future scenarios given by a scenario tree, this condition translates into the equation:

$$\sum_j \mathbf{p}_j^{(i)} \cdot \mathbf{z}_j^{(\mathrm{a}(i))} >= \Phi_T^{(i)}, \quad \forall i \in \mathcal{T}.$$

As a full coverage of the target payoff is strictly required, a super-replication model does not require the decomposition of the error variable into the positive and negative parts ($\boldsymbol{e}_i^+$ and $\boldsymbol{e}_i^-$) as presented in the previous model.

Managing to meet the condition imposed by this model may be very challenging and often this reflects in an excessively elevated cost for the strategy: in order to mitigate this aspect, it is usual to set the minimization of an objective function represented by the total cost of the hedging. Assuming a self-financing strategy without possibility to withdraw capital after an initial investment, the problem solved at stage 0 can be formulated as:

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n} \left[(1+\mathbf{c}_j)\cdot \mathbf{p}_j^{(0)} \cdot \mathbf{x}_j^{(0)} - (1-\mathbf{c}_j)\cdot \mathbf{p}_j^{(0)} \cdot \mathbf{y}_j^{(0)}\right] && (3.10)\\
\text{s.t.} \quad & \mathbf{x}_j^{(0)} - \mathbf{y}_j^{(0)} = \mathbf{z}_j^{(0)}, \quad \forall j && (3.11)\\
& \mathbf{z}_j^{(\mathrm{a}(i))} + \mathbf{x}_j^{(i)} - \mathbf{y}_j^{(i)} = \mathbf{z}_j^{(i)}, \quad \forall i \in \mathcal{N}, \forall j && (3.12)\\
& \sum_{j=1}^{n} (1-\mathbf{c}_j)\cdot \mathbf{p}_j^{(i)} \cdot \mathbf{y}_j^{(i)} - \sum_{j=1}^{n} (1+\mathbf{c}_j)\cdot \mathbf{p}_j^{(i)} \cdot \mathbf{x}_j^{(i)} = 0, \quad \forall i \in \mathcal{N} && (3.13)\\
& \sum_j \mathbf{p}_j^{(i)} \cdot \mathbf{z}_j^{(\mathrm{a}(i))} - \Phi_T^{(i)} >= 0, \quad \forall i \in \mathcal{T}. && (3.14)\\
& \mathbf{x}_j^{(i)}, \mathbf{y}_j^{(i)} \geq 0, \quad \forall i \in \mathcal{N} \cup \{0\}, \ \forall j && (3.15)
\end{aligned}$$

No further details will be given on problems solved at subsequent decision stages, as for the purpose of this thesis only the problem at decision stage 0 will be considered. Constraints (3.11)-(3.13) and (3.15) are the same as the previous model. What differs is the objective function and the definition of the super-replication constraint.

Objective Function. In the super-replication hedging problem, the goal of the objective function is to minimize the total cost of the hedging while ensuring the super-replication of the target payoff at maturity as in Equation 3.14. Since the strategy is assumed to be self-financing, the only cost faced corresponds to the capital invested in the portfolio construction at the root node of the scenario tree; thus, for all assets, the total amount of cash invested in buying asset holdings minus the capital gained from asset sales is added to the objective function in order to account for the wealth invested in the portfolio and impose its minimization.

Super-replication Constraint. The constraint given by Equation 3.14 imposes that, for each leaf node $i \in \mathcal{T}$ in the scenario tree, the hedging portfolio's value

$$\sum_j \mathbf{p}_j^{(i)} \cdot \mathbf{z}_j^{(\mathrm{a}(i))}$$

must be at least equal to the target payoff $\Phi_T^{(i)}$. This constraint is the key of the super-replication problem, ensuring that the portfolio's value exceeds the liability due to the target payoff in every simulated scenario.

Liquidity Fund and Replication Tolerance

As anticipated, meeting the super-replication constraint given by Equation 3.14 can be challenging. In particular, there might be scenarios within the simulated paths with a high value for the target payoff, deviating from the average behavior; to guarantee that the hedging portfolio achieves a future value sufficient to offset the liability across all scenarios, the strategy may require an excessively high initial investment for the portfolio construction. To deal with this issue, one choice may be to allow for a higher degree of flexibility in meeting the super-replication condition: in this context the liquidity fund plays a key role.

The liquidity fund, denoted as L_f, represents the hedger tolerance threshold with respect to deviations of the hedging portfolio value from the derivative's payoff, meaning that the portfolio value is allowed to fall short of the target payoff by no more than L_f. The incorporation of this term in the super-replication constraint could mitigate the issue related to excessively high initial investments, leading to a

more reliable solution. In light of what discussed, Equation 3.14 would become:

$$\sum_j \mathbf{p}_j^{(i)} \cdot \mathbf{z}_j^{(\mathrm{a}(i))} - \Phi_T^{(i)} + L_f >= 0, \quad \forall i \in \mathcal{T}.$$

The decision to include the liquidity fund in the constraint, as well as the choice of its value, depends on the hedger, according to his available wealth.

Chapter 4

Key aspects of the implementation

Across the previous chapters and sections, some aspects of the practical implementation of the hedging strategy have already been examined. For example, the generation of scenario trees has been discussed along with the different dynamics assumed (one based on the Geometric Brownian Motion, the other employing the Moment Matching method), as well as the two approaches tested to check the absence of arbitrage opportunities in simulated scenario trees. The financial market setup was also defined, specifying the implemented assets formulas, and the various alternative formulations considered for the hedging problem have been detailed.

Before moving on to the discussion about the performance of the proposed hedging approach, in addition to what was previously described, some additional key aspects of the implementation are of particular interest and will be detailed in the course of this chapter. Particular attention will be given to the simulation process with its specific implementation, outlining the role of Monte Carlo simulation and the structure given to the simulations of the test phase, which recalls the general paradigm of Reinforcement Learning. Further considerations on the implementation will also be provided, including the selection of model parameters by the user when interacting with the developed code, a description of the UML diagram of financial instrument classes and additional discussions on the two following topics: the management of branching factors in scenario tree generation and the structure given to the cash balance constraint in the optimization problem.

For reference, the project underlying the implementation of the hedging strategy proposed in this thesis has been developed in Python programming language.

4.1 Simulation process

A first discussion about implementation details concerns the mechanism under which the developed hedging strategy is tested in the financial market environment. This section will examine the methodology of the test phase for the hedging strategy, highlighting the generation of simulated paths for the market evolution and how the hedger interacts with the financial environment during his decision-making process.

4.1.1 Monte Carlo Simulation

Monte Carlo simulation represents, more than just a method, a general paradigm to simulate a wide range of stochastic dynamics, analyze the behavior of stochastic variables of particular interest and estimate their statistical properties with high accuracy.

The core idea underlying a Monte Carlo simulation is inspired by the **law of large numbers**: when a progressively larger number of samples is generated from the same probability distribution, the sample mean converges to the true mean of the distribution and the precision of the estimate progressively improves. Briefly, let $(X_1, \ldots, X_n)$ be independent and identically distributed random variables from a distribution with mean μ and variance σ^2: according to the law of large numbers, as n approaches infinity, the sample mean $\bar{X}$ converges almost surely to the true mean μ and the variance of the estimator (that is, $\sigma^2(\bar{X})$) decreases proportionally to $1/\sqrt{n}$, implying that the variance in the estimate decreases as the number of Monte Carlo replications n increases.

Following the idea of the law of large numbers, the objective of Monte Carlo simulation is to infer the expected behavior of a system by averaging over a large number of simulated paths. Indeed, by generating a set of replications of the stochastic variables' evolution, when the average of some statistical properties computed on the set of sampled paths converges to a stable value, the simulation has, on average, approximated the true system behavior.

In the context of the present thesis, Monte Carlo simulation plays a key role in evaluating the performance of the hedging strategy. A large number of replications of the financial market evolution is simulated and the hedging strategy is applied to each of them; this methodology ensures that the estimate of the hedging performance becomes progressively more reliable and accurately reflects the true effectiveness of the proposed hedging strategy.

4.1.2 Test with the Reinforcement Learning paradigm

The test of the hedging strategy in the financial market environment has been implemented with a simulation process following the Reinforcement Learning (RL) paradigm. For each Monte Carlo replication in which the hedging strategy is applied, the following structure to the test process is given: at each decision stage, the hedging agent observes the current market state and determines the optimal portfolio rebalancing decision; the environment then makes a step forward to the next market state according to the realization of stochastic variables. This iterative process continues until the target asset's maturity is reached.

To clarify the structure of Monte Carlo simulations, refer to Figure 4.1, where the interaction between the hedging agent and the financial market environment is illustrated as a pseudo-code.

Hedging strategy simulation following the Reinforcement Learning paradigm

```
def run_simulation(env, HedgingAgent, branching_factors):
    env.reset() reset the environment to start a new MC simulation
    for rep in range(num_MonteCarlo_reps):
        returns the market state at stage 0
        state = env.restart()
        while not done:
            returns the branching factors for the current time step
            bf = branching_factors(state)

            the agent makes the portfolio rebalancing decision
            action, cost = HedgingAgent.get_action(state, bf)

            update the portfolio according to the agent's decision
            update_portfolio(env, action)

            returns the market state at the following timestep
            state = env.step()
```

Figure 4.1: Pseudo-code of the simulation process inspired by the Reinforcement Learning paradigm.

In order to give a hint on the mechanism under the *get_action* function, Figure 4.2 is provided, detailing the decision-making process of the agent after observing the current market state.

HedgingAgent.get_action(state, branching_factors)

```
def get_action(state, branching_factors):
    Generate a scenario tree to simulate assets prices paths
    Tree = ScenarioTree(branching_factors, stoch_model,
                        current_asset_values)

    Determine the optimal portfolio rebalancing strategy
    Hedging = StochOptHedgingSolver(Tree, current_time,
                                    target_asset, hedging_assets)

    Store the portfolio holdings after rebalancing
    portfolio_holdings = store_holdings(Hedging)

    Extract the rebalancing cost
    rebalancing_cost = Hedging.rebalancing_cost

    Return the portfolio and rebalancing cost
    return portfolio_holdings, rebalancing_cost
```

Figure 4.2: Pseudo-code of function get_action to determine the optimal hedging strategy given the market state and a simulated scenario tree.

4.1.3 Out-of-sample paths generation

The assumption that the hedger's decisions do not influence market transitions is made: asset prices follow stochastic processes and the sequence of their realizations is not affected by the transactions of the individual hedger, who, in fact, does not have any impact on the financial market dynamics. This assumption, which is quite common in financial applications (particularly for individual hedgers), allows to simulate a set of asset price paths without requiring, in advance, knowledge of the hedger's decisions at each stage; this allows to separate the simulation logic of the market evolution and the logic of the hedger decision-making process,

facilitating the implementation of the whole test mechanism. However, an important aspect to take into account is that the hedging strategy implemented must be *non-anticipative*: despite the set of previously simulated price paths available, the hedger should rebalance the portfolio at each time t based on the information given by the filtration $\mathcal{F}_t$ and not having access to the realizations of future asset prices.

4.2 Additional considerations

Hedging configuration

The hedging strategy described in chapter 3 includes a multitude of variables and parameters and, at the same time, admits different choices for the hedging strategy configuration to implement. To better manage the configuration and the parameters selection, an interactive interface has been implemented in order to collect the user's choices about the inputs of the problem, as shown in Figure 4.3.

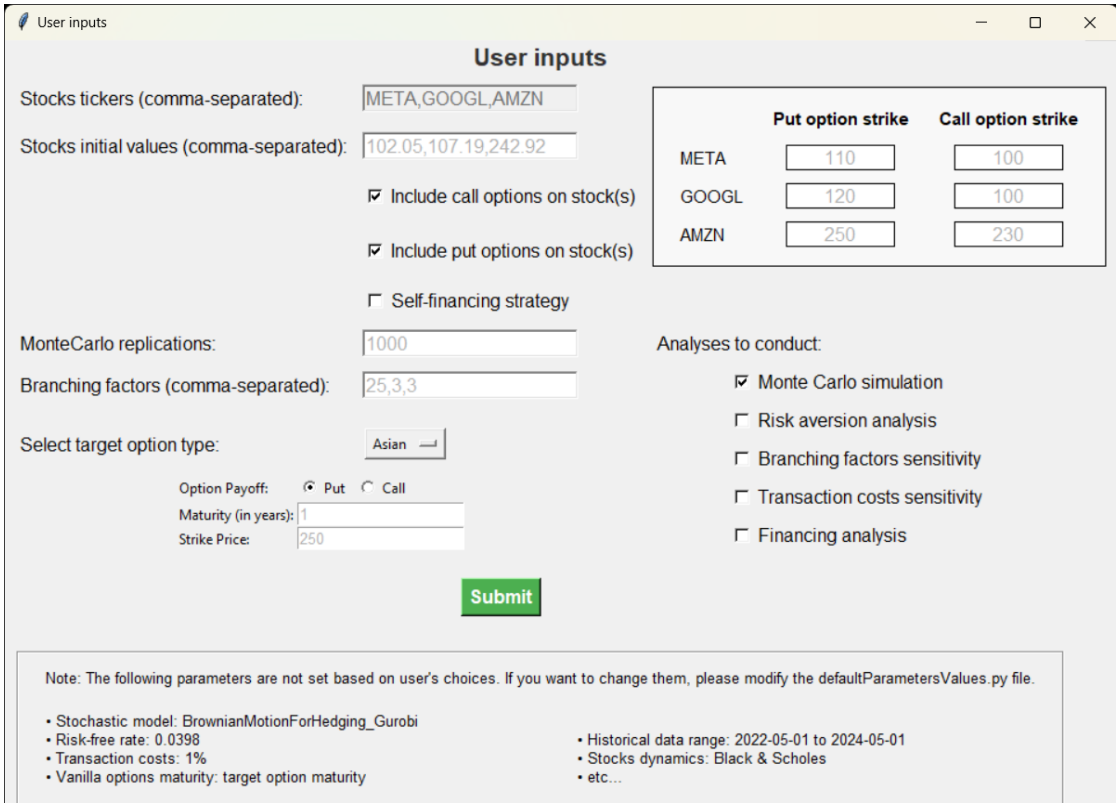


Figure 4.3: Interactive interface for the selection of user inputs.

As shown, default values for all the parameters have been set in order to allow the user to modify only the parameters of his interest. Among the others, the interface

allow for customized choices for the hedging instruments to include in the strategy, as well as the choice for the target option to hedge (with each option type featuring its set of parameters). Further flexible choices are left for the number of Monte Carlo replications, the branching-factors for scenario tree simulation, whether to apply a self-financing strategy or not and the analysis to conduct (these will be better described in the following chapter). Not-mentioned parameters and hedging configurations might also be modified directly in the python file.

Branching factors updating strategy

When talking about scenario trees, one aspect of particular interest is the management of branching factors along the hedging horizon: if at stage 0 the choice for the branching factors falls, for instance, on the vector [25,3,3], one might wonder how the branching factors should be chosen for the next time steps, that is, when we move forward at subsequent stages in the dynamic decision process after the first decision is made. Different approaches may be considered:

- implement a *forward erosion* updating strategy, according to which branching factors for the sequence of the three decision stages would be:

  $$\text{stage } 0 : [25,3,3] \rightarrow \text{stage } 1 : [3,3] \rightarrow \text{stage } 2 : [3].$$

  This choice turns out to be not effective, since the choice of branching factors should always prioritize the first branching steps with higher branching factors for an effective performance (as it will be illustrated in section 5.5);

- implement a *backward erosion* updating strategy, according to which branching factors would be:

  $$[25,3,3] \rightarrow [25,3] \rightarrow [25].$$

  Eroding the last entrance of the vector, at each decision stage a higher number of final scenarios in the scenario tree is maintained with respect to the previous strategy. This is the main approach adopted in the applications presented in the following chapter;

- implement a *fixed number of scenarios* updating strategy, according to which branching factors would be:

  $$[25,3,3] \rightarrow [75,3] \rightarrow [225].$$

  This updating strategy allows to maintain, at each decision stage, the same number of leaf nodes as in the scenario tree generated at stage 0; a comparative analysis of this strategy with respect to the previous one will be conducted, for now it is sufficient to note that this is not the most frequently chosen approach

in this work due to its higher computational burden and the fact that it does not lead to significant improvements in terms of performance when the first entry of the vector is already a high number (as in the considered case, where from the root node 25 child nodes branch out).

Modeling dependencies among asset classes

In the project implementation, the development of asset classes follows an object-oriented approach, resulting in a hierarchical structure that leverages the concepts of inheritance and relationship. In fact, considering both assets in the hedging portfolio and the target asset to be hedged, although each asset exhibits distinct characteristics, they share some common attributes and methods and present some reciprocal interactions.

The interdependencies among asset classes have been structured according to the UML diagram shown in Figure 4.4. This diagram illustrates the relationships between the considered asset classes, starting from the abstract *Asset* class, which represents a parent class for other financial instruments, to its three primary subclasses: *Cash*, *Stock* and *Option*, each class modeling the correspondent financial instrument. The *Option* class itself is also defined as an abstract class, further implemented by the *EuropeanOption* and *AsianOption* subclasses (of both put and call payoffs). In addition, a *MultiStock* class is taken into account, useful to link all the stocks available in the market through their correlation matrix ρ. Through this hierarchical structure, each subclass inherits attributes and methods from its parent class, while also introducing its own attributes and methods specific to the type of asset it models.

Cash Balance constraint: clarifications

In chapter 3, various alternatives for the hedging problem were introduced, each employing a different portfolio financing strategy: recall, for instance, the choices between a self-financing or a non-self-financing strategy, the possibility to allow (or not allow) cash withdrawal from the hedging portfolio and the decision to set a fixed initial wealth invested in the portfolio or to leave the solver the possibility to choose the best suited initial wealth.

In light of these different alternatives, to better clarify the structure given to the cash balance constraint of Equation 3.4, applied at the root node of a scenario tree in the hedging problem, a Python pseudo-code of the practical implementation of this constraint under different financing assumptions is shown in Figure 4.5.

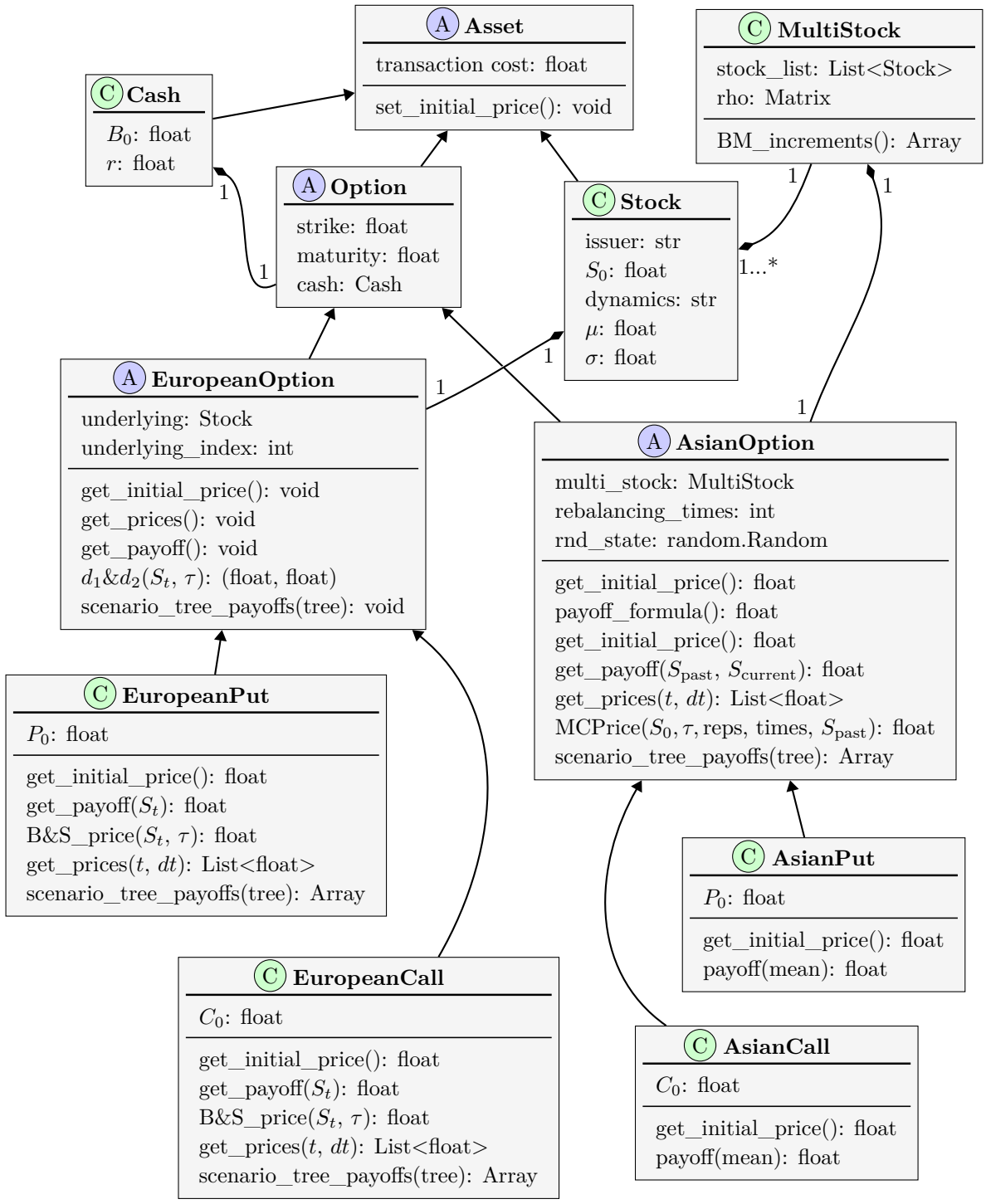


Figure 4.4: UML diagram of asset classes.

```
# Root node of the scenario tree
if not withdraw_possibility:  # No withdraw possibility
    if current_rebalancing_time == 0:
        # Cash balance at stage 0
        if self.fixed_wealth: # Fixed wealth
            model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost
            + target_asset_premium == 0)
        else: # Non-fixed wealth
            model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost <= 0)
    elif not self_financing:
        # Not self-financing strategy or at the first stage
        model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost <= 0)
    else:
        # Self-financing strategy
        model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost == 0)
else:
    # There is withdrawal possibility
    if current_rebalancing_time == 0:
        # Cash balance at stage 0
        if self.fixed_wealth: # Fixed wealth
            model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost
            + target_asset_premium == 0)
        else:  # Non-fixed wealth
            model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost <= 0)
    elif self_financing:
        # Subsequent stages with a self-financing strategy
        model.addConstraint(
            - purc_quantity @ purc_unitcost
            + sold_quantity @ sold_unitcost >= 0)
    else:
        # Subsequent stages, not self-financing: no constraint
        pass
```

Figure 4.5: Pseudo-code of the cash-balance constraint of the hedging optimization problem under different financing assumptions.

Chapter 5

Performance analyses

In this chapter, the objective is to evaluate the overall performance of the proposed stochastic optimization approach for the hedging strategy by analyzing its effectiveness under multiple aspects, including computational burden, stability and, above all, precision in risk minimization. To highlight the effectiveness of the method, the different shades of the problem implementation presented in chapter 3 are examined in what follows.

The analysis starts from a general consideration over a Monte Carlo simulation of the stochastic optimization hedging; in this phase, a first analysis of the hedging performance is inferred by testing the reliability of the hedging solver over a high number of simulated paths for the financial marker evolution. Both European and Asian options are considered as target assets, willing to compare the performance and the efficiency of the strategy when facing different target option types.

The chapter proceeds then through analyses on the performance of the hedging under different problem-configurations: these include comparing different portfolio financing approaches, analyzing the impact of the risk-aversion parameter on the solution of the hedging problem, analyses on branching factors sensitivity, which shows the impact on the solution of the scenario tree generation and its discretization of the future scenarios space, transaction costs sensitivity and the analysis of the performance of the hedging strategy under different stochastic models used in generating scenario trees. In a final step, a discussion about how the hedging problem can be used to price financial derivatives is made.

5.1 Performance evaluation metrics

Before entering into details of the hedging performance, some clarifications and definitions should be made about the evaluation metrics that will be used.

Hedging Error and Profit&Loss

During the course of this chapter, two key evaluation metrics are considered for the analysis of the hedging strategy: the Hedging Error (HE) and the Profit&Loss (P&L). The former was already mentioned in previous chapters, while the latter is still to be defined.

Recall that the hedging error is defined as the difference, at maturity T, between the hedging portfolio value V_T^h and the target asset payoff Φ_T, that is

$$\text{HE} = V_T^h - \Phi_T.$$

The Profit&Loss is a more comprehensive measure of the total balance of gains and losses incurred throughout the hedging process.

Let us make an example of an hedging strategy with a specific focus on cash flows: refer to the situation described by Figure 5.1. At the initial time, the

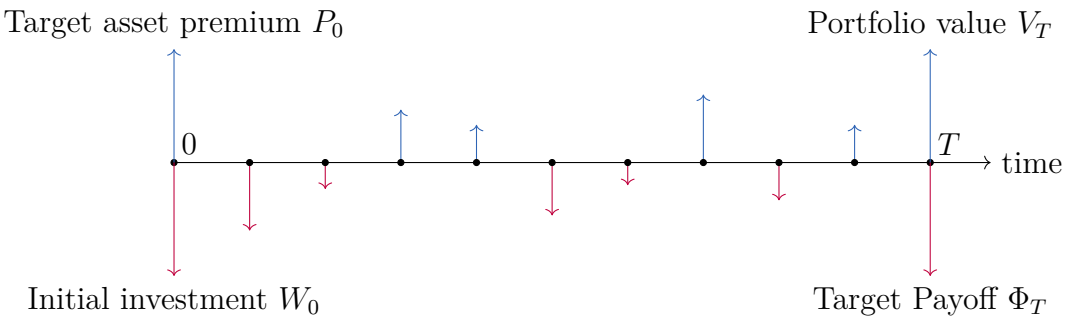


Figure 5.1: Example of - positive and negative - cash flows for a hedging strategy.

target asset premium is gained by shorting the position in the target asset, while a negative cash flow is due to the portfolio construction; along the hedging horizon various cash flows may occur: negative cash flows are due to further financing operations of the portfolio, while positive cash flows are related to the withdrawal of cash from the strategy. At maturity T, according to the market scenario that occurs, a liability may be encountered due to the target payoff Φ_T, while selling the hedging portfolio generates a positive cash flow.

In light of this general situation, the Profit&Loss (P&L) for a hedging strategy is defined as the sum of all cash flows capitalized to maturity, that is:

$$\text{P\&L} = \sum_{t=0}^{T} c_t \cdot e^{r(T-t)}$$

where c_t is the cash flow at time t. Trivially, the P&L corresponds to the hedging error for a self-financing strategy with an initial wealth equal to the target asset premium and no possibility of withdrawal. However, in other situations, the two may differ and the P&L should be preferred as the primal evaluation metric as it includes additional information with respect to the hedging error.

What was just described is the evaluation framework for a **hedged position**; trivially, for an unhedged position, the hedging error is formally not defined; the P&L, instead, is the difference between the target asset premium received at the initial time actualized at maturity T and the target payoff Φ_T, that is:

$$\text{P\&L}_{\text{unhedged}} = P_0 \cdot e^{rT} - \Phi_T$$

5.2 Monte Carlo simulation of the hedging

In this section, a first look at the performance of the Stochastic Optimization hedging strategy described in previous chapters is given. The hedging strategy is implemented for both vanilla put and call European options and put and call Asian options, illustrating the effectiveness of the algorithm developed in minimizing risks related to the the short position in each of the mentioned target option. In particular, hedging error and Profit&Loss will be used as metrics to evaluate the performances.

Regardless of the target option selected, the hedging framework for this section will be configured with the following features:

- $n = 5000$ Monte Carlo replications;
- hedging horizon of $T = 1$ year with 4 time steps and 3 decision stages (no portfolio rebalancing occurs at maturity date);
- branching factors [25,3,3], with the backward erosion strategy for their update at subsequent stages (that is, branching factors [25,3,3] $\rightarrow$ [25,3] $\rightarrow$ [25] to respectively generate scenario trees at stages 0, 1 and 2);
- the set of available stocks composed of the following tickers:

  - ENEL.MI, with initial price \$102.05;
  - MMM, with initial price \$242.92;
  - TSLA, with initial price \$107.19;

- parameters for stocks' dynamics $(\mu, \sigma, \rho, \text{skew}, \text{kur})$ estimated from a set of historical data in the range from 05/01/2022 to 05/01/2024;
- proportional transaction costs equal to 1% of the asset price for each risky asset (transaction cost 0% for the risk-free asset);
- risk aversion parameter γ set to the value 1;
- the strategy configured as self-financing and without possibility to withdraw money along the hedging horizon (thus, in this situation, the hedging error correspond to the P&L);
- the initial wealth set equal to the target asset premium, thus each Monte Carlo replication have the same initial budget for the portfolio construction;
- the annualized continuous risk-free rate assumed to be $r = 0.0398$.

Concerning the verification of absence of arbitrage opportunities in scenario trees, the problem defined by Equation 2.9 is exploited, but this specific choice (between the two approaches mentioned in subsection 2.2.3) was not driven by any specific reason: indeed, preliminary tests pointed out that both methods effectively ensure the absence of arbitrage opportunities, while featuring comparable computational costs (in truth, one approach proved to be more efficient than the other in certain cases and viceversa for other cases, thus there is no strict rule to justify the choice of one method against the other one).
In addition to the given configurations, each target asset will then present specific parameters that will be added to those already mentioned.

5.2.1 Hedging European options

Two Vanilla European option are selected for the implementation of the hedging strategy, respectively one European call option with underlying ENEL.MI, strike price $K = 50$ and one European put option with underlying TSLA and strike price $K = 300$, both with maturity $T = 1$ year. When dealing with target European options, the hedging portfolio is composed of the underlying stock and the risk-free asset, denoted as Cash.
Figure 5.2 and Figure 5.3 show the empirical distribution of the Profit&Loss for both the hedged and the unhedged position over the set of simulated Monte Carlo replications, respectively, for the European call and the European put.

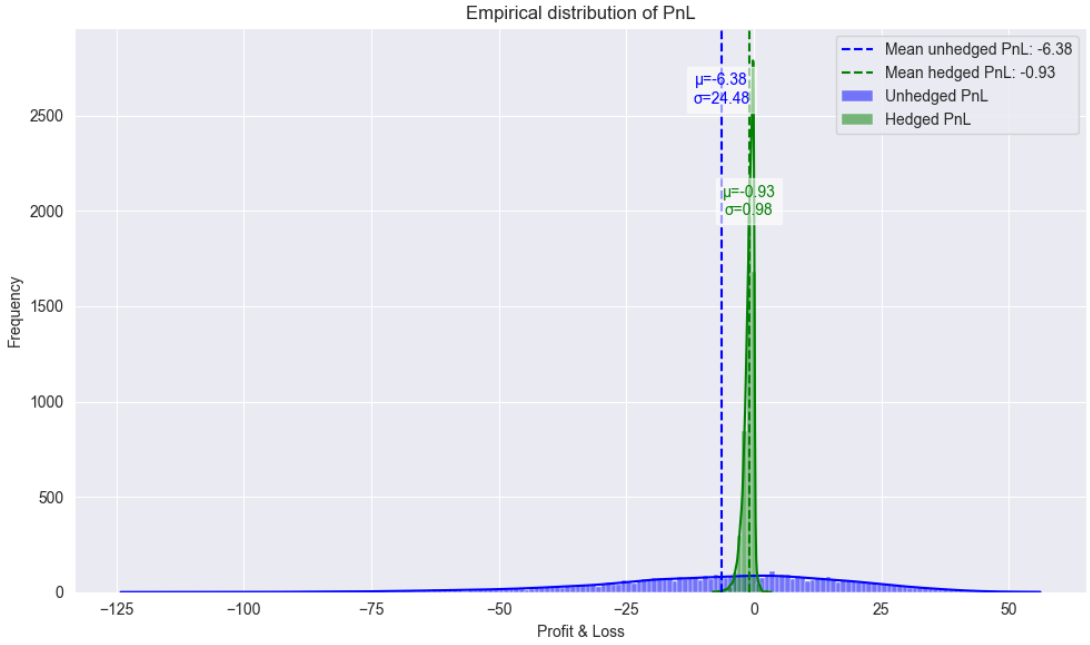


Figure 5.2: Empirical distribution of P&L for an European call option.

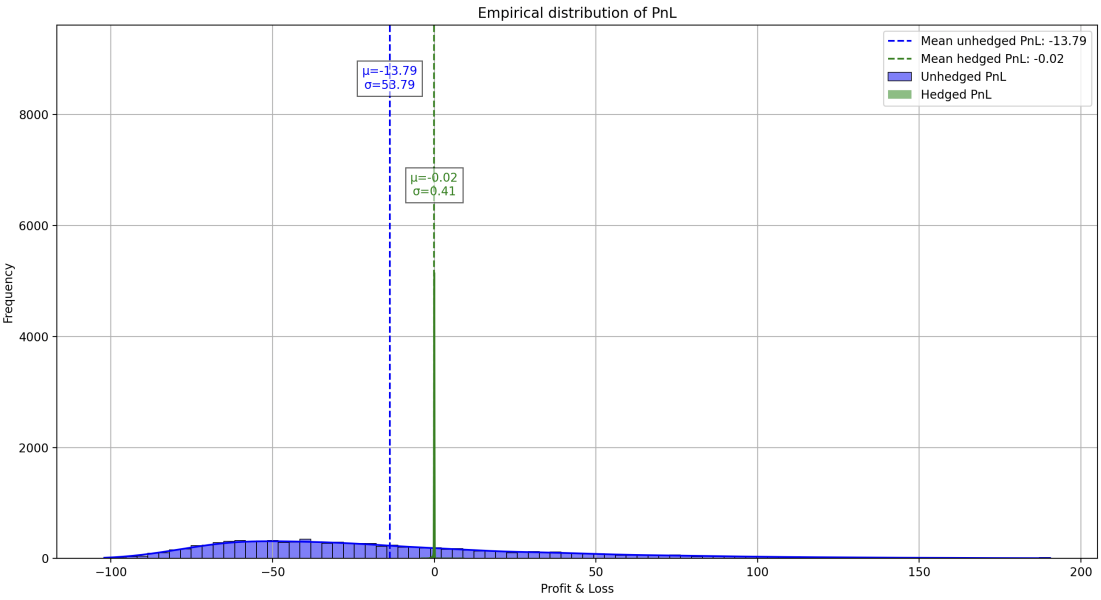


Figure 5.3: Empirical distribution of P&L for an European put option.

The strategy managed to hedge the short position in both vanilla options. The P&L of the hedged position features an empirical distribution highly concentrated

around the mean, with the latter assuming values close to 0 for both the call and put options, against the high variance of the unhedged position.

5.2.2 Hedging Asian options

The Asian options implemented to test the strategy are written on the whole set of available stocks introduced before, the Asian put option with strike price $K = 250$, while the Asian call with strike $K = 50$, both with maturity $T = 1$ year. Strategies for both options include the following hedging assets in the portfolio: the risk-free asset (cash position), the set of underlying stocks and both call and put vanilla options on the underlyings (one call and one put for each stock); in particular, Table 5.1 summarizes the strike prices for each vanilla option on the underlyings.

		stocks		
target asset	vanilla option	ENEL.MI	MMM	TSLA
Asian call	European put	130	260	120
	European call	70	200	90
Asian put	European put	170	300	200
	European call	100	200	90

Table 5.1: Strike prices for vanilla options on the available stocks.

Figure 5.4 and Figure 5.5 show the empirical distribution of Profit&Loss over the set of simulated Monte Carlo scenarios for, respectively, the Asian call and the Asian put. Similarly to what was obtained for European options, the effectiveness of the hedging is confirmed by the lower variance in the P&L distribution of both Asian call and Asian put with respect to the variance in the unhedged positions, with mean values for the P&L close to 0.

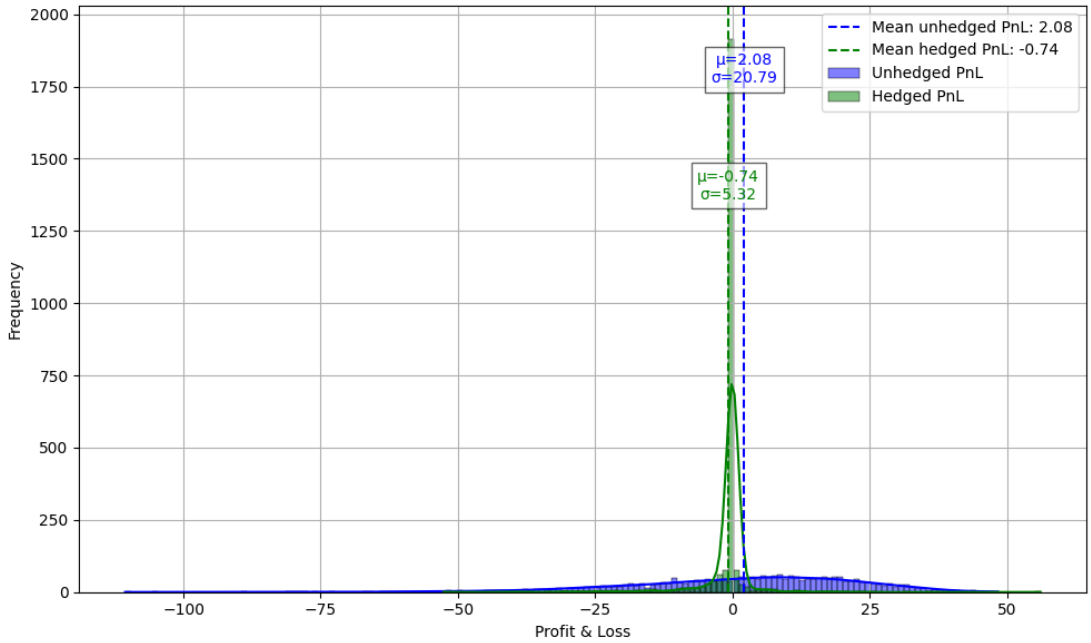


Figure 5.4: Empirical distribution of P&L for an Asian call option.

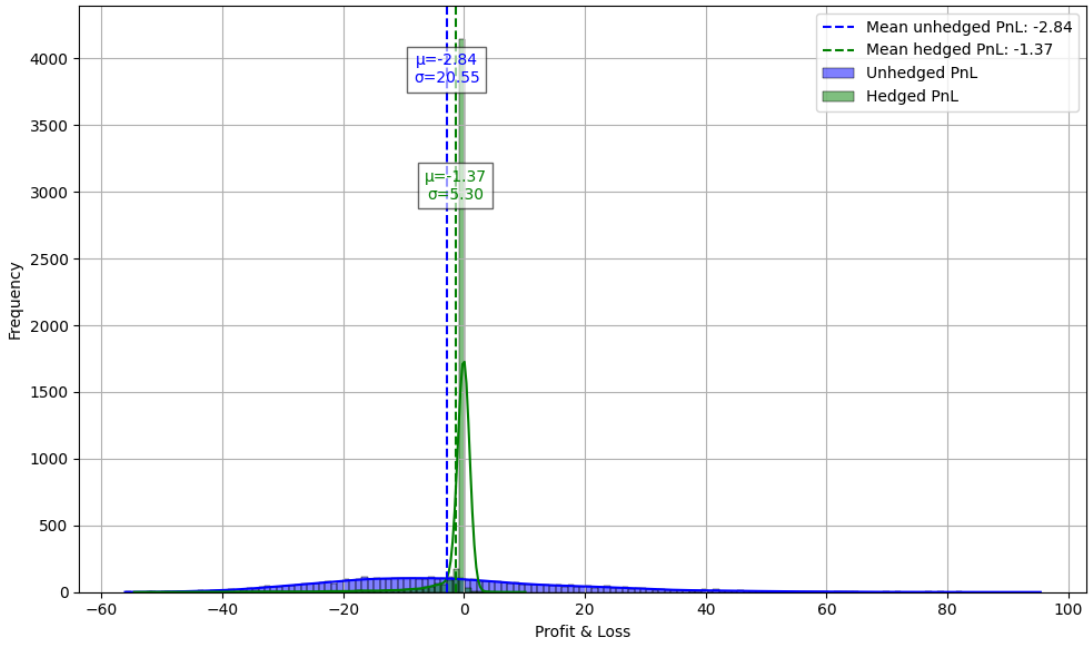


Figure 5.5: Empirical distribution of P&L for an Asian put option.

5.2.3 Comparative analysis of different target option types

Table 5.2 summarizes the performance obtained from the first analysis conducted for European and Asian options.

	mean P&L	std P&L	std unhedged	computational time
European call	0.93	0.98	24.48	1232.20 s
European put	-0.02	0.41	53.79	1084.82 s
Asian call	0.74	5.32	20.79	5647.03 s
Asian put	1.37	5.30	20.55	4328.54 s

Table 5.2: Summary of statistics for European and Asian options hedging for a MC simulation over 5000 replications.

Statistics from this first analysis point out that the hedging strategy is capable of managing both vanilla options, as the European ones, and exotic options, such as the Asian ones; mean values for the P&L obtained are similar for the two different option types, while a slight difference appears in the values of standard deviation, with exotic options featuring higher values compared to the European ones. Moreover, the computational time required to complete the Monte Carlo simulation over 5000 replications shows a difference from this comparative analysis; however, both increases in standard deviation and computational time for Asian options may be due to the presence of a wider range of hedging assets in the portfolio, increasing the complexity of the optimization problem solved at each decision step.

Another aspect of interest in the comparative analysis between European and Asian options concerns the effectiveness of the target replication step by step. In P&L distributions previously provided, the attention is focused only at maturity date, comparing the unhedged position against the hedged one at the end of the hedging horizon. Willing to analyze the hedging behavior also at intermediate time steps, step-by-step replication plots are now provided to show the behavior of the hedging strategy for both European and Asian options along the hedging horizon. Figure 5.6 shows the replication of the target asset at each decision stage; in particular, for each time step, the following are provided:

- the hedging portfolio value;
- the value of the unhedged position, representing the value of the target asset: this is represented by the option price, except for the final stage (corresponding

to the maturity) where the payoff is provided instead; this value is changed in sign as a short position in the target asset is assumed;

- the value of the hedged position, representing the sum of the first two terms mentioned.

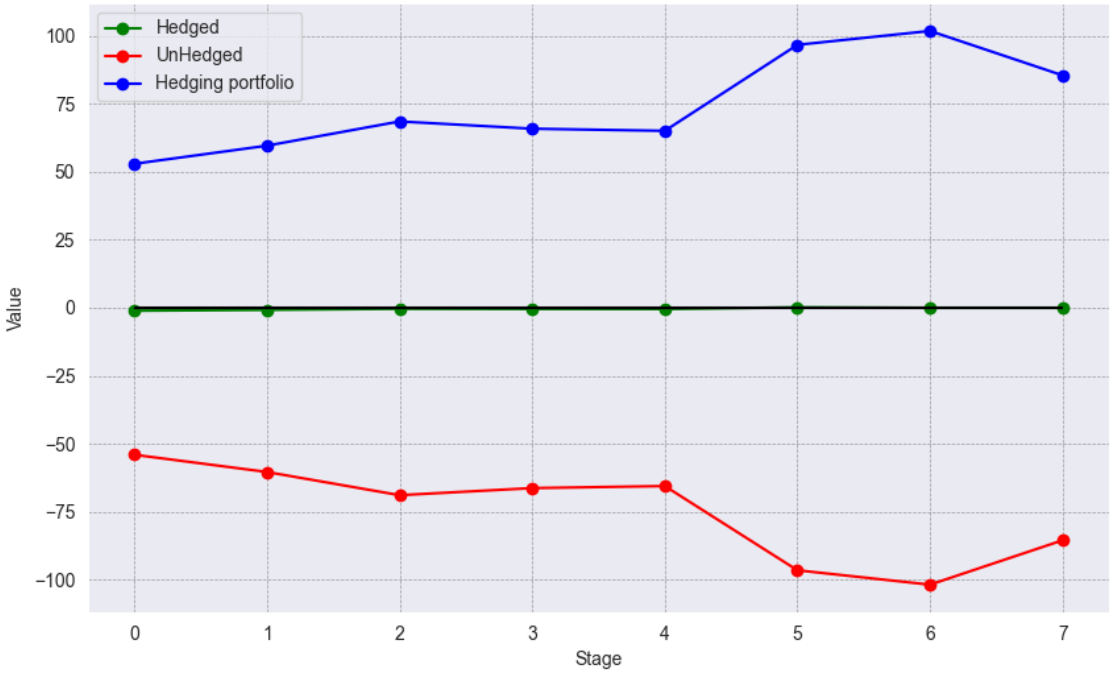


Figure 5.6: Step-by-step replication of an European call option in 1 Monte Carlo simulation.

The figure shows how precisely the hedging portfolio tracks the European option value, resulting in a hedged position whose value is kept around 0 along the whole hedging horizon.

Concerning the hedging of Asian options, Figure 5.7 provides the same plot for an Asian put option. The concept is the same for the previous case with an European option, although a less accurate replication is encountered in some intermediate stages, probably due to the higher complexity in managing an exotic option. This discrepancy between the target asset value and the hedging portfolio value is more evident for the plot of Figure 5.8, which again illustrates the replication of the same Asian put option, but for a different Monte Carlo simulation. It is clear from this latter figure that a less effective step-by-step replication of the target option value is obtained compared to the previous case; however, an accurate hedging of the target payoff at maturity is achieved, which is the main objective of our strategy.

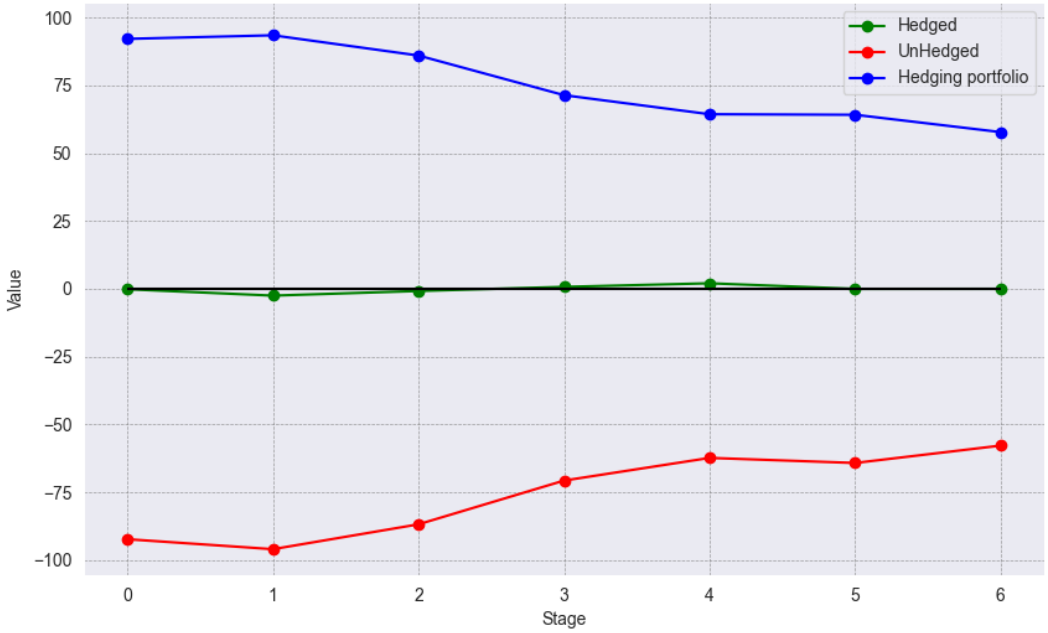


Figure 5.7: Step-by-step replication of an Asian put option in one Monte Carlo replication.

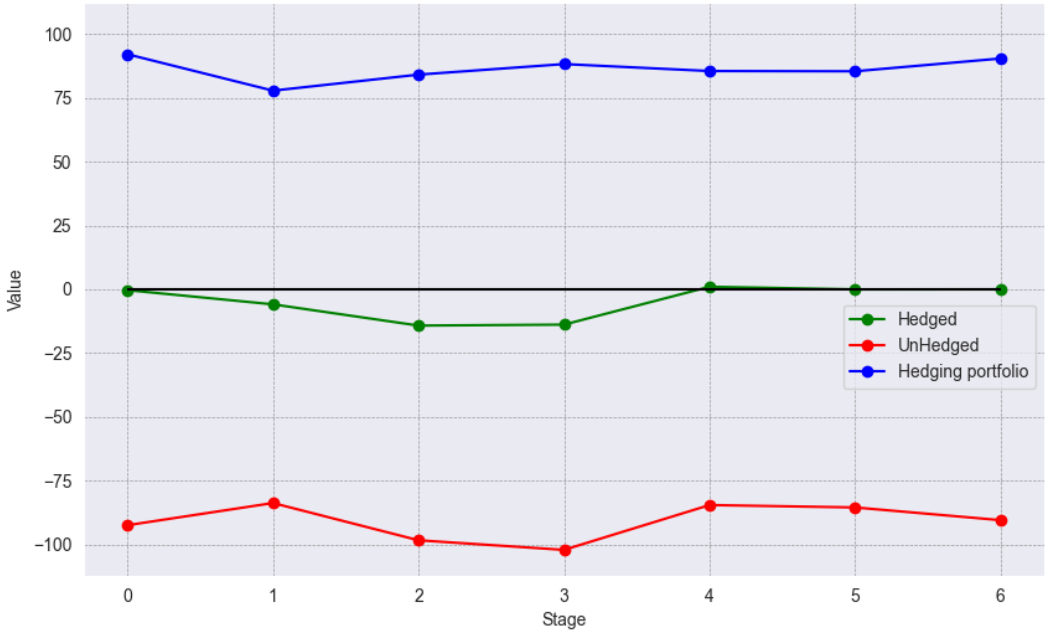


Figure 5.8: Another example of step-by-step replication of an Asian put option in one Monte Carlo replication.

Marginal Consideration. Concerning what was just discussed, it should be recalled that the optimization model solved at each step aims to hedge the payoff at maturity; any perfect replication along the hedging horizon results from effective intermediate rebalancings. Obtaining a step-by-step replication of the target asset is promising, as it indicates that the hedging strategy is proceeding well and facilitates perfect replication at maturity; however, these intermediate replications are not explicitly required by the optimization and their absence does not compromise the success of the strategy.

5.3 Risk aversion analysis

In this section, an analysis of the impact of the risk-aversion parameter γ on the hedging performance is conducted. In particular, for different values of $\gamma \in [0,1]$ a Monte Carlo simulation of the hedging is performed and an estimate of the profit and loss (P&L) is computed for each γ.

Experimental Setup. The chosen values for γ belong to the set:

$$\gamma \in \{0, 10^{-13}, 10^{-12}, 10^{-11}, \cdots, 10^{-3}, 10^{-2}, 10^{-1}, 1\}.$$

This choice aims at showing different responses of the hedging strategy with respect to a increasing risk-neutrality condition, starting from the value $\gamma = 0$, reflecting the most risk-averse scenario in which only hedging shortfalls are penalized, and reaching progressively the value $\gamma = 1$, under which there is a symmetrical penalization of positive and negative hedging errors.
For each value of the parameter γ, a Monte Carlo simulation of the hedging over $n = 10^3$ replications is performed and an estimate for the P&L is computed by averaging over the number of replications. The hedging is configured as a self-financing strategy with an initial wealth fixed to the target asset premium and without possibility of withdrawal; thus, in this situation, the P&L corresponds to the hedging error.

Results. Figure 5.9 shows mean and standard deviation values of the empirical distribution of P&L over the simulated Monte Carlo replications for each value of the risk-aversion parameter. Additionally, a hint on the value of the skewness for these distributions is given by the asymmetry of the standard deviation around the value given by the mean. As expected, the figure highlights a positive drift in P&L as γ approaches 0. However, it is not the behavior of the mean which is of greater interest, as it increases slowly, but rather the progressive increase in standard deviation and skewness, which points out that the P&L distributions exhibit heavier upper tails as γ decreases: thus, for small values of the risk-aversion

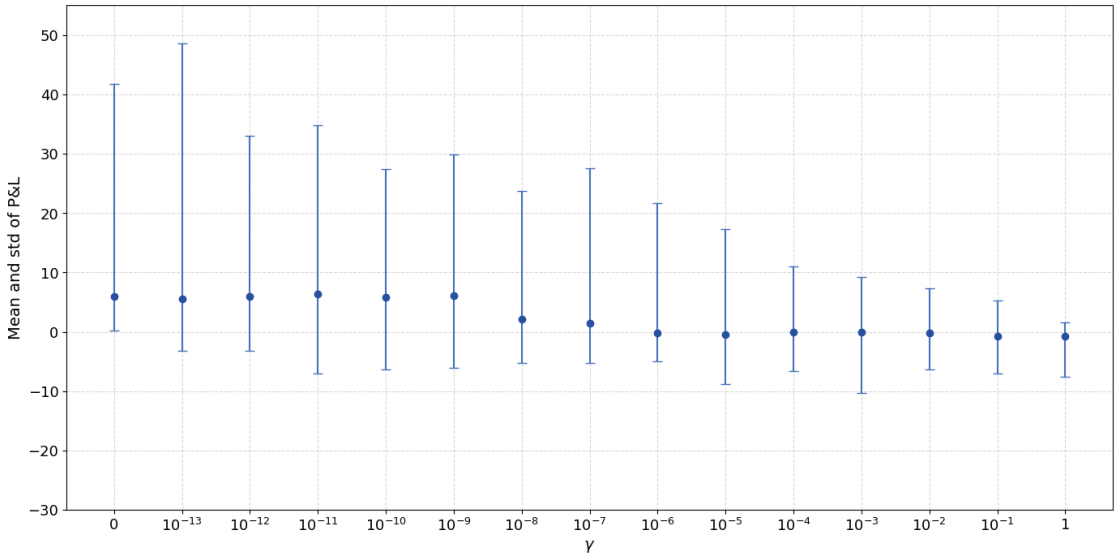


Figure 5.9: Mean and standard deviation of P&L empirical distribution over 10^3 simulated paths for each different value of the risk-aversion parameter.

parameter, P&L distributions are asymmetrical and shifted towards higher values (as mentioned before, this behavior is due to the minor importance given to surplus values in the hedging error).
Table 5.3 reports the values of the mentioned statistics for some values of the risk aversion parameter, confirming the increasing trend in standard deviation and skewness observed in the figure.

γ values:		1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-12}	0
P&L	mean	-0,79	-0,22	-0,09	-0,27	2,06	5,82	5,93	5,95
	std	4,55	6,86	8,79	13,33	14,46	16,89	18,06	20,76
	skew	-4,47	0,99	2,36	5,84	4,48	2,56	4,52	6,60

Table 5.3: Mean, standard deviation and skewness of P&L empirical distribution for different values of the risk aversion parameter γ.

Another interesting aspect related to γ concerns the initial wealth W_0: knowing that for small values of γ the hedging strategy almost ignores positive hedging errors focusing primarily on covering negative hedging errors, it could result in a surprisingly high initial investment required, if the latter is not fixed; in fact, if the surplus at the maturity date is not penalized, the model finances the portfolio with a higher initial wealth in order to ensure that negative errors are covered to a

significant degree. To visualize this, another Monte Carlo simulation is conducted over $n = 10^3$ simulated paths for each value of γ; the strategy is similar to the one described before, except for the absence of a fixed initial wealth. Figure 5.10 shows the results of this simulation, pointing out, for different values of γ, the mean and standard deviation (along with a hint on the skewness) of the empirical distribution of initial investment over the simulated replications. The figure confirms the

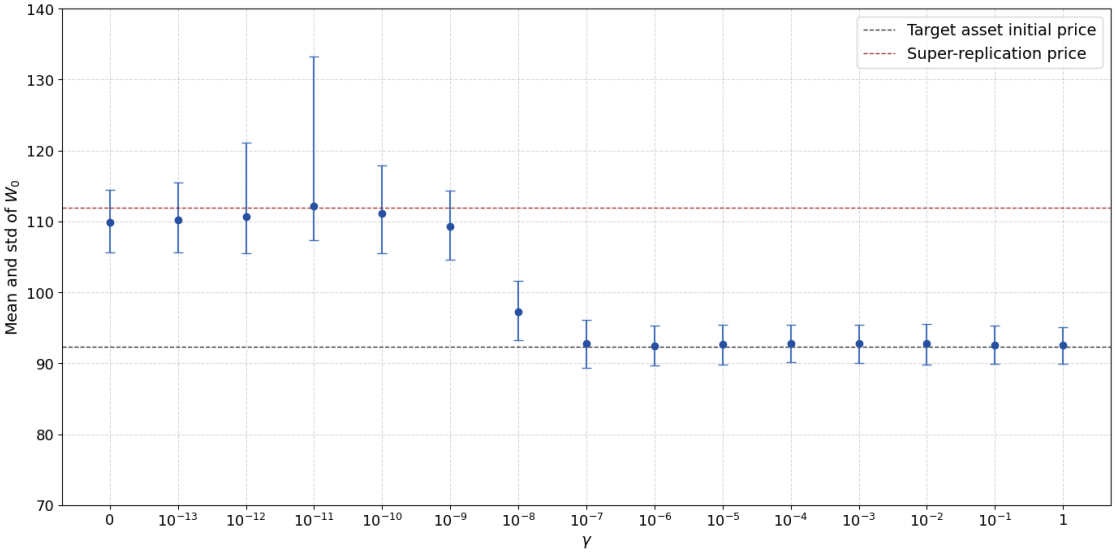


Figure 5.10: Mean and standard deviation of the empirical distribution of W_0 over 10^3 simulated paths for different values of the risk-aversion parameter. The mean super-replication price for the same MC simulations is also provided.

expectation: when there is an asymmetrical penalization of surplus and shortfall errors, the initial investment is shown to be higher, both in terms of mean value and standard deviation. For γ close to 1, the initial investment required by the hedging strategy reflects quite accurately the initial price of the target asset, while for γ approaching 0 the mean initial wealth progressively increases and its distribution features heavier upper tails. To provide a reference, in the figure is also shown the super-replication price (corresponding to corresponds to \$111.91) computed by averaging over the simulated MC scenarios the initial wealth required by solving a super-replication hedging problem; it can be seen that for an asymmetrical penalization of the positive and negative part of the hedging error, the mean value for the wealth W_0 gets close to the super-replication price (recall that the super-replication problem ensures a perfect hedging against negative hedging errors in simulated scenarios, disregarding positive hedging errors, while minimizing the total cost of the hedging). For a better understanding of this behavior, Table 5.4 is also provided, showing mean, standard deviation and skewness of the empirical

distribution of initial investment for different γ values.

γ values:		1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-12}	0
W_0	mean	92,54	92,76	92,79	92,50	97,31	111,17	110,72	109,91
	std	2,57	2,88	2,62	2,81	4,17	6,18	7,81	4,40
	skew	0,01	-0,39	0,27	-0,09	0,66	2,11	8,21	0,83

Table 5.4: Mean, standard deviation and skewness of the empirical distribution of initial wealth W_0 required by the strategy for different values of the risk aversion parameter γ.

5.4 Financing analysis

As introduced in chapter 3, the hedging strategy may be either self-financing or not and may admit the possibility to withdraw an amount of capital from the portfolio. These conditions influences not only how the portfolio is financed but also the hedging performance. In this section, an analysis of the impact of the financing strategy on the hedging performance is conducted, in order to give a hint on the possible configurations of the hedging strategy and their effectiveness.

Experimental Setup. A Monte Carlo simulation is performed for each combination of parameters self-financing $\in$ {True, False} and withdrawal-possibility $\in$ {True, False}. The key implementation points of this analysis include an Asian put option as the target asset (the one described in section 5.2), $n = 10^3$ Monte Carlo replications for each financing configuration, initial wealth fixed and equal to the target asset premium, 4 decision stages along the hedging horizon and branching factors equal to [6,4,3,2] (with the fixed number of scenarios updating strategy, thus with a fixed total of 144 scenarios also for scenario trees generated at later stages).

The self-financing strategy without possibility to withdraw cash has already been implemented and studied in section 5.2; in what follows, performances of other combinations of self-financing and withdrawal possibility will be detailed.

Results. For a non-self-financing strategy without possibility to withdraw, after an initial investment used to construct the hedging portfolio, further financing is allowed at following decision stages along the hedging horizon. Cash flows for

additional financing are assumed to be negative, indicating that the hedger loses an amount of available cash which is invested in the portfolio. Figure 5.11 shows the empirical distribution of P&L for this configuration, while Table 5.5 shows mean and standard deviation of cash flows at intermediate stages; these statistics are also provided for the total cash flow, the latter representing the sum over decision stages of the cash flows. Additionally, the rebalancing frequency for each stage is provided, computed as the number of Monte Carlo simulations in which an additional financing happened at that specific stage over the total number of simulations. Note that decision stages at time 0 and T have trivially cash flows equal to zero, as the initial wealth corresponds to the target asset premium (thus they offset each other) and at maturity date no rebalancing operation is planned; thus these stages are not reported in the table.

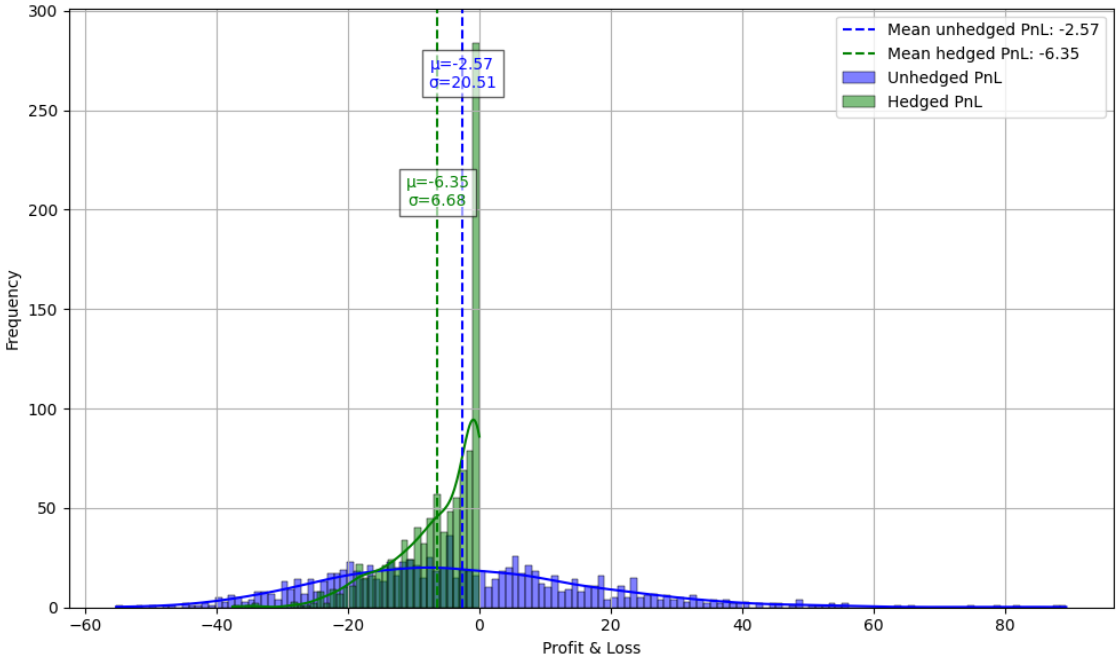


Figure 5.11: Empirical distribution of P&L for a non-self-financing strategy without withdrawal possibility.

Results for this implementation of the hedging problem shows good performances in terms of hedging error (which features a mean equal to -0,001 with a standard deviation of 0,0323, pointing out that the replication of the target payoff was achieved), but the additional financing during intermediate stages lead to a higher variance in P&L distribution (as shown in the figure); thus, the non-self-financing condition makes the hedging more accurate but with an increased variability in P&L compared to the self-financing strategy.

	decision stages			
	stage 1	stage 2	stage 3	total
mean cash flow:	-3,793	-1,797	-0,602	-6,192
std of cash flows:	5,020	2,917	1,836	6,512
financing frequency:	66,9%	53,3%	45,6%	

Table 5.5: Statistics of cash flows at intermediate stages for a non-self-financing strategy without withdrawal possibility.

On the other hand, for a self-financing strategy with possibility to withdraw, after an initial investment used to construct the hedging portfolio, money can be withdrawn from the portfolio at subsequent decision stages; cash flows for this setting are assumed to be positive, indicating that the hedger gains an amount of money which is withdrawn from the the hedging strategy. As for the previous configuration, Figure 5.12 shows the P&L distribution for this strategy, while Table 5.6 shows mean and standard deviation of cash flows at intermediate stages, along with the frequency at which cash withdrawal operations happen.
Also for this financing configuration, the hedging error distribution is satisfactory (mean equal to -2,046 with standard deviation 6,382), but the impact of withdrawing cash from the strategy at rebalancing stages lead to a higher variance in the P&L distribution, making this financing strategy less effective than the already mentioned ones.

	decision stages			
	stage 1	stage 2	stage 3	total
mean cash flow:	3,083	0,945	0,466	4,495
std of cash flows:	5,263	2,527	2,233	6,487
withdrawal frequency:	52,33%	30,97%	16,70%	

Table 5.6: Statistics of cash flows at intermediate stages for a self-financing strategy with possibility to withdraw.

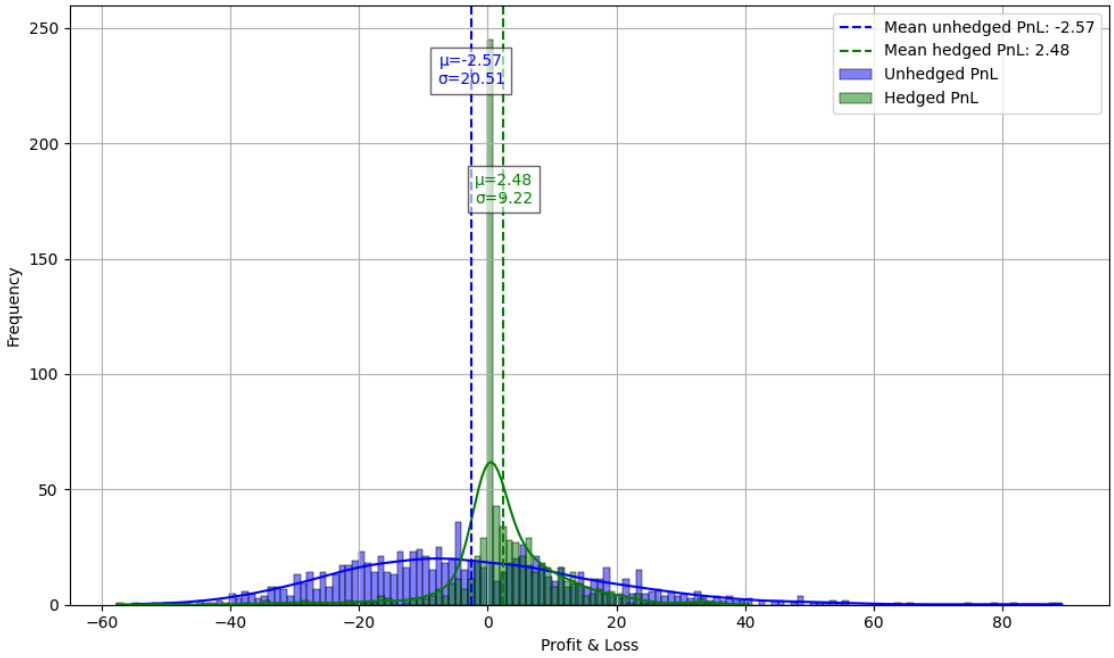


Figure 5.12: Empirical distribution of P&L for a self-financing strategy with possibility to withdraw money.

Lastly, the non-self-financing strategy with possibility to withdraw cash from the hedging portfolio is the strategy which combines the characteristics of the two previous approaches, allowing both positive and negative cash flows at intermediate stages. Table 5.7 is provided with statistics about cash flows at intermediate stages.

	decision stages			
	stage 1	stage 2	stage 3	total
mean cash flow:	-3,238	-3,158	-0769	-6,993
std of cash flows:	9,586	7,536	4,576	12,641
cash flows frequency:	100%	100%	100%	

Table 5.7: Statistics of cash flows at intermediate stages for a non-self-financing strategy with possibility to withdraw money.

On average, the solver requires additional cash to finance the strategy rather than allowing to withdraw part of the portfolio wealth; one aspect of particular interest

is the cash flows frequency: indeed, for this configuration it turns out that in every intermediate stage (of all MC replications) occur a cash flow, thus there is a frequency of cash flows equal to 100% for each intermediate stage.
This configuration, like the others, performed well in terms of hedging error (mean: 0.002; std: 0.248), but it was the least effective in terms of P&L (whose distribution features mean: -7.95; std: 14.40).

Discussion. It is clear from the different configurations implemented that, when cash flows occur, they are most likely to happen during the first rebalancing stages, while as the maturity date approaches cash flows become less frequent and involve smaller amounts of money. Results also show that intermediate cash flows, whether positive or negative, make the strategy less effective, leading to improvements in the hedging error distribution but not significant enough to offset the increase in variance of the P&L distribution (recalling that P&L is what the hedger is more interested in). To better visualize this (also with the reference of the self-financing strategy without withdrawal possibility implemented in a previous section), Table 5.8 summarizes the statistics for hedging error and P&L distributions for all the financing configurations.

	HE		**P&L**	
	mean	std	mean	std
self-financing: T, withdrawal: F	1,37	5,30	**1,37**	**5,30**
self-financing: F, withdrawal: F	**-0,001**	**0,03**	-6,35	6,68
self-financing: T, withdrawal: T	-2,05	6,38	2,48	9,22
self-financing: F, withdrawal: T	0,002	0,248	-7,95	14,40

Table 5.8: Summary of statistics for hedging error (HE) and Profit&Loss (P&L) for different financing strategies.

5.5 Branching factors sensitivity

In this section, an analysis of sensitivity of the hedging performance with respect to the choice of branching factors (denoted as bf for simplicity) is conducted; through this, a discussion is made about how different scenario tree configurations impact on both solution quality and computational time. A Monte Carlo simulation of the hedging with $n = 10^3$ replications for each selected configuration of branching factors is performed in order to give a first hint on the best choice of branching factors. The hedging configuration corresponds to the one described in section 5.2 for a target Asian put option, except for not fixing the initial wealth, letting the hedging solver choose the best-suited one.

One aspect of particular interest is the management of out-of-sample test scenarios, which are generated a-priori; considering that, for a fixed configuration of the hedging strategy, the length dt of time-steps is generally computed as

$$dt = \frac{\text{target option maturity}}{\text{number of decision stages}} = \frac{T}{\text{length(branching factors)}},$$

to generate out-of-sample scenarios that are the same for all the bf configurations, the length dt is chosen to be the least common multiple ($l.c.m$) among the lengths of time-steps individuated by the bf configuration tested; each hedging strategy then exploits the value of dt corresponding to the actual time step individuated by its branching factors (e.g. the hedging configuration with [10,5,4] uses $dt = \frac{T}{3}$).

Table 5.9 illustrates the results of this analysis. In particular, for each choice

bf	**nodes**	**time**	**HE**		**cost**		**P&L**	
			mean	std	mean	std	mean	std
[10,5,4]	260	111.6	−1.69	16.00	93.41	2.06	−2.78	16.08
[25,6,5]	925	804.4	−0.62	5.67	93.22	0.50	−1.51	5.70
[25,15]	400	74.42	0.07	5.54	93.89	0.56	−1.52	5.56
[3,3]	12	3.79	−13.10	150	85.37	33.69	−5.82	151.56
[4,4,4,4]	340	197.36	−0.34	72.16	94.13	7.42	−2.17	72.28
[25,5,5,5]	3900	13540.67	−0.711	6.92	106.25	25.35	−15.93	27.05
[25,5,4,3]	2150	8039.81	−1.01	7.14	92.12	4.06	−0.71	8.32

Table 5.9: Summary of statistics for the branching factor analysis, pointing out the sensitivity of the hedging performance to the choice of branching factors.

of bf, the following are reported: the total number of nodes in the scenario tree

(corresponding to the number of nodes for the tree at decision stage 0, while trees at subsequent stages will have progressively less nodes since the backward erosion strategy is assumed), the computational time of the MC simulation and values of mean and standard deviation for: hedging error (HE), cost of the hedging strategy and Profit&Loss; these statistics are computed by averaging over the set of MC replications. In order to provide a reference, the unhedged position features for this set of MC simulations a P&L empirical distribution with mean -4.281 and standard deviation 21.56; this is true for all the branching factor configurations, as the simulated out-of-sample scenarios are the same for every bf.

Results discussion. The results from the table lead to some considerations of particular interest:

1. for a more effective hedging, configurations with higher branching factors for the first stages, while progressively decreasing for subsequent stages, should be preferred to configurations with homogeneous branching factors. The evidence lies in the behavior of the two configurations [4,4,4,4] and [10,5,4]: the former has homogeneous branching factors, while the latter has branching factors homogeneous by time-step and higher for the first stage. Even if [4,4,4,4] features a higher number of nodes and time-steps, configuration [10,5,4] achieves the best accuracy, particularly for values of standard deviation in P&L distribution;

2. for the bf updating strategy with backward erosion, that is for example

   $$[6,5,4,3] \rightarrow [6,5,4] \rightarrow [6,5] \rightarrow [6],$$

   choosing a small value for the branching factor at the first stage is not convenient: in that case, the scenario tree generated at the last decision stage would not generate enough scenarios to approximate properly the final outcomes space; for those configurations, it is better to choose a higher value for the first entrance of the bf vector, that is for example

   $$[20,5,4,3] \rightarrow [20,5,4] \rightarrow [20,5] \rightarrow [20],$$

   or rather implement the fixed number of scenarios updating strategy, according to which trees for all decision stages have the same number of leaf nodes; this latter strategy would be:

   $$[6,5,4,3,2] \rightarrow [12,5,4,3] \rightarrow [36,5,4] \rightarrow [144,5] \rightarrow [720];$$

3. when comparing the configurations [25,5,4,3] with [25,5,5,5], it can be observed that the latter has a higher computational time which is not justified by the improvement in results; the former configuration requires less computational burden while ensuring a satisfactory result.

Minor considerations might also be done. The results shown in the table indicate that, as expected, for higher values of branching factors a heavier computational burden is encountered: this highlights the growth in complexity when generating trees with higher number of nodes. For example, branching factors [25,5,5,5], which feature a number of nodes equal to 3900, are those with the highest computational time to complete the MC simulation, that is 13540.67 s; on the other hand, the scenario tree with branching factors [3,3], which features only 12 nodes, requires 3.789 s to complete simulation of the hedging.

In terms of hedging error and P&L, the configuration [25,15] achieves the best performance with a moderate computational time (74.42 s): for this configuration there is an effective trade-off between hedging accuracy and computational time required; the configuration [3,3], instead, points out that too simplistic choices of bf fail to provide a reliable solution to the hedging problem. If the performances for the two configurations are compared, it can be concluded that the not-reliable solution of [3,3] is not due to the low number of rebalancing stages, but rather to the limited number of tree nodes, as already commented.

Lastly, concerning the hedging cost (equal, for this analysis, to the initial portfolio construction, as a self-financing strategy is assumed), given that the Monte Carlo price of the target put option is 92.367, almost all configurations of branching factors present satisfactory results, with few exceptions: this means that almost all configurations required an initial wealth close to the target asset initial price. The only exceptions are:

- configuration [3,3], which underestimated the target asset premium, probably due to the scenario tree being too small to efficiently approximate the space of possible future outcomes;

- the [25,5,5,5] configuration, which, on the other hand, overestimated the target asset premium; this overestimation is directly reflected in the poor-quality results in terms of P&L empirical distribution, which is worse compared to the other cases.

A further analysis involves studying the behavior of different branching factors configuration with the fixed number of scenarios updating strategy, that is, maintaining a fixed the number of leaf nodes for scenario trees at different stages. A total of $n = 10^3$ Monte Carlo replications are performed for the branching factors [25,3,3] and [10,5,3] in order to analyze this updating strategy and benchmark the results against those previously obtained, as reported in Table 5.10. For this implementation, a constant initial wealth (equal to the target asset premium) is considered, thus hedging error and profit and loss coincide. Contrary to what

	Number of leaf nodes			time	P&L	
	1st stage	2nd stage	3rd stage		mean	std
[10,5,3]	150	50	10	100.66 s	−0.712	6.57
	150	150	150	168.48 s	−1.008	3.36
[25,3,3]	225	75	25	236.09 s	−0.885	4.59
	225	225	225	433.67 s	−0.873	3.38

Table 5.10: Comparative analysis between the backward erosion updating strategy for branching factors (gray) and the approach with a constant number of leaf nodes in scenario trees (blue).

was expected, the reported table shows that the updating strategy with a fixed number of leaf nodes in scenario trees has not led to significant improvements in P&L distribution; rather, the only notable difference with respect to the previous approach regards its computational time, which is trivially increased due to the higher complexity of scenario trees at subsequent stages. The fact that this approach has not led to enough improvements to justify its increased computational time may limit its practical implementation in favor of the previous strategy.

5.6 Transaction costs analysis

As anticipated in previous chapters, the Stochastic Optimization approach implemented for the hedging problem may take into account the presence of transaction costs when trading financial instruments, allowing to formulate a framework of asset-trading which is more consistent with real-world financial markets. In this thesis, proportional transaction costs are assumed, that is, denoting by x_j the quantity of asset j traded at price p_j, the cost for trading asset j is:

- $(1 - c_j)x_j p_j$ if the asset is purchased;
- $(1 + c_j)x_j p_j$ if the asset is sold.

In this section, an analysis of the impact of transaction costs on the hedging performance is conducted, pointing out how the hedging solver responds to different configurations of transaction costs. The optimization setting is the same as the one depicted in section 5.2 (Asian put option with strike 250, maturity 1 and a strategy which is self-financing with fixed initial wealth and no withdrawals allowed).

The tested values of proportional transaction costs belong to the set:

$$\{0,10^{-10},10^{-8},10^{-6},10^{-4},10^{-2},10^{-1}\}.$$

It is assumed that every asset shares the same value of transaction cost, except for the risk-free asset, which is assumed to be traded without any additional costs. The results of this analysis are analyzed in term of P&L empirical distribution, as shown in Figure 5.13; in particular, the mean and standard deviation of P&L over the simulated MC replications are given. Surprisingly, the figure points out

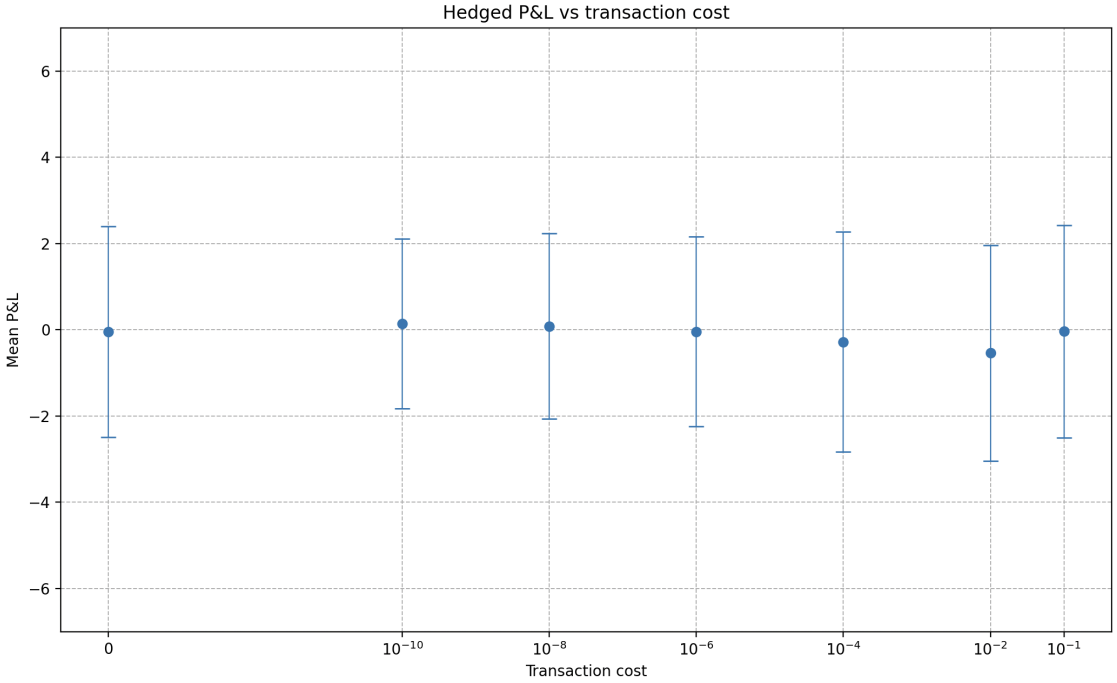


Figure 5.13: Analysis of sensitivity of the hedging performance (in terms of mean and standard deviation of P&L empirical distribution) with respect to different values of proportional transaction costs incurred in assets trading.

that the implemented hedging solver seems not to be influenced by the choice of transaction costs, achieving a good quality solution in term of P&L regardless of the transaction costs applied to trading operations. Indeed, all the P&L show a mean close to 0 with a low and stable value of standard deviation, thus ensuring an effective reduction in risk exposure even for cases with high transaction costs (such as, 10%). Additionally, transaction costs proved to have neither impact on the computational time required to solve the hedging problem: this result highlights the strength of the hedging problem solved through the stochastic optimization approach proposed.

5.7 Scenario tree generation: GBM vs MM

In subsection 2.2.1 and subsection 2.2.2 two different approaches for scenario tree generation were introduced: the simulation via Geometric Brownian Motion and the simulation via Moment Matching. In this section, a brief comparative analysis is conducted in order to evaluate their effectiveness, both for computational time and hedging performance.

The hedging configuration of this analysis exploits an Asian put option with strike $K = 250$ and $T = 1$ year to test the performance of the three approaches to scenario tree generations; the hedging strategy is assumed to be self-financing, without withdrawal possibility and with an initial wealth fixed to the target option premium.

Figure 5.14, Figure 5.15 and Figure 5.16 show the hedged P&L empirical distribution for, respectively, the implementation of the Geometric Brownian Motion with SLSQP, the Geometric Brownian Motion with Gurobi and the Moment Matching approach; to provide a reference, the unhedged position for these simulations features a standard deviation of 11.26.

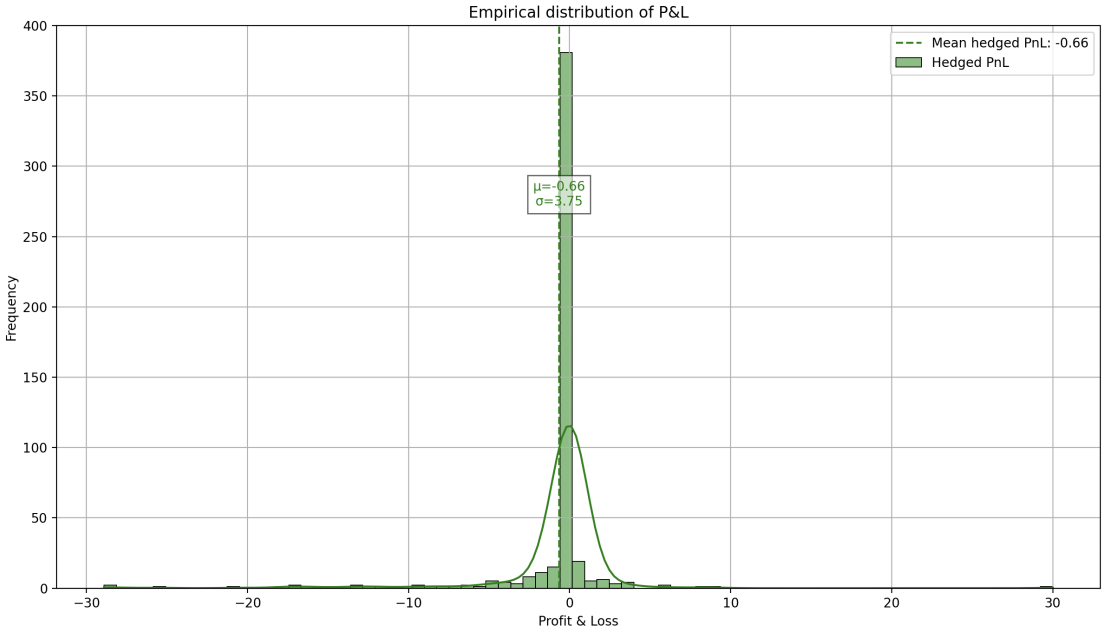


Figure 5.14: Profit&Loss empirical distribution for the implementation of GBM − SLSQP in scenario tree generation.

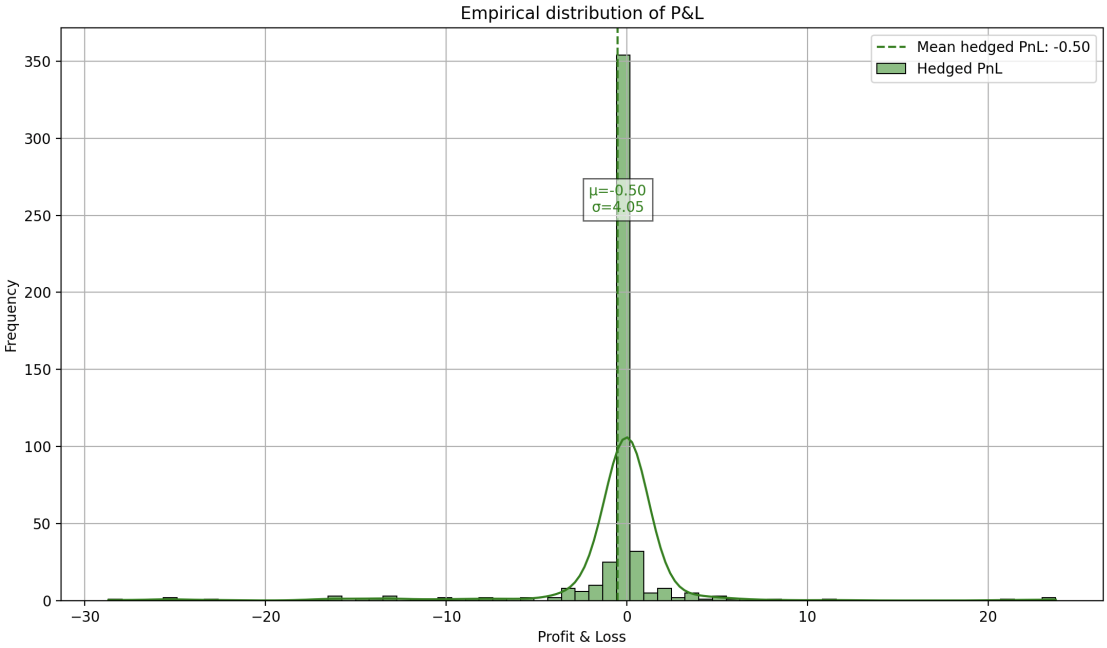


Figure 5.15: Profit&Loss empirical distribution for the implementation of GBM − Gurobi in scenario tree generation.

The figures evidence that all three implementations lead to satisfactory results, hedging the position in the target asset and improving the variability of the unhedged position (which is recalled to have a standard deviation of 11.26).

Despite the good performance of all the stochastic models, further discussion can be provided. Concerning the computational time, the Moment Matching turned out to be the most time-consuming, with a total of 4423.73 s required to complete the hedging simulation, against the 746.18 s of the implementation of GBM with SLSQP and the 277.85 s of the GBM implementation with Gurobi. Thus, GBM with Gurobi turned out to be the approach with the lowest computational time, outperforming the SLSQP alternative, although the feature a comparable P&L distribution.

However, despite being the slowest approach, the Moment Matching has some key advantages: including the 3^{rd} and 4^{th} moments in the set of matched properties, it proved to lead to some improvements in the hedging accuracy even in hedging configurations which are not so promising, as when using branching factors [15,5,4], which features a limited number of branches in scenario trees.

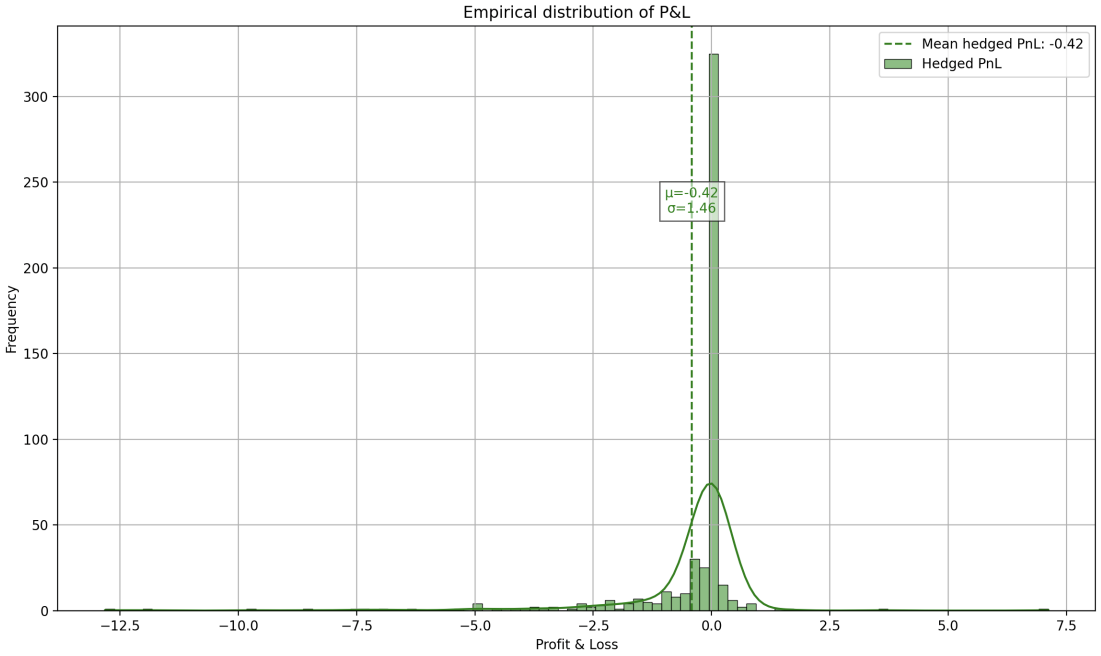


Figure 5.16: Profit&Loss empirical distribution for the implementation of Moment Matching in scenario tree generation.

Summarizing, Table 5.11 reports the discussed performances of the three approaches to scenario tree generation.

	P&L		**time**
	mean	std	
GBM - Gurobi	-0.50	4.05	277.85 s
GBM - SLSQP	-0.66	3.75	746.18 s
Moment Matching	-0.42	1.46	4423.73 s

Table 5.11: Summary of performance statistics for different approaches to scenario tree generation.

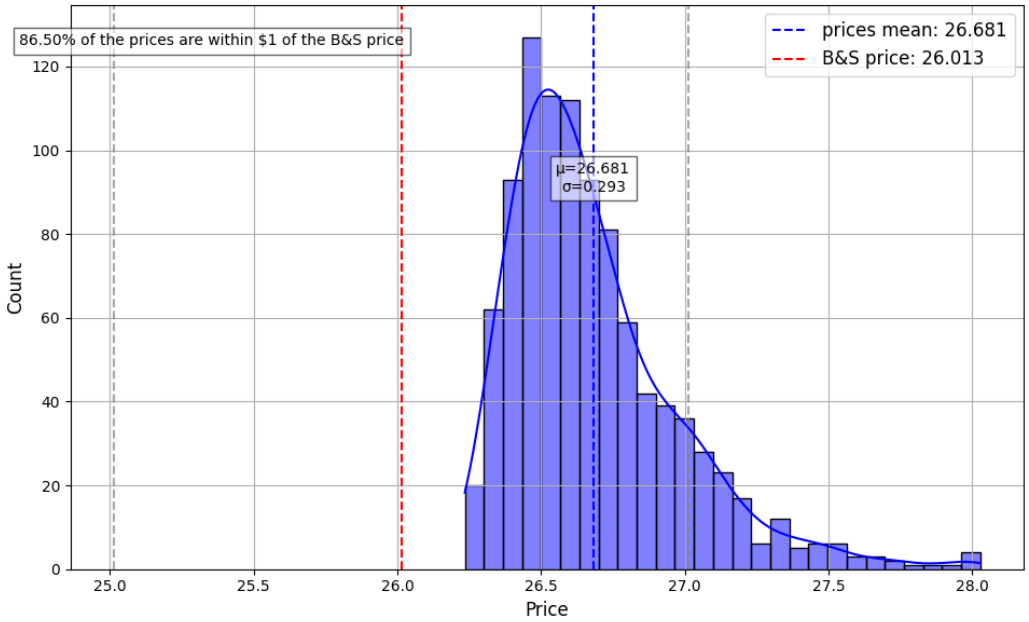
This analysis highlighted the effectiveness of all the implemented approaches in modeling scenario trees while also showing some of their strengths and weaknesses related to hedging accuracy and computational burden. Clearly, based on the analysis conducted, none of these approaches should be excluded a priori: the choice should be left to the hedger.

5.8 Pricing through hedging

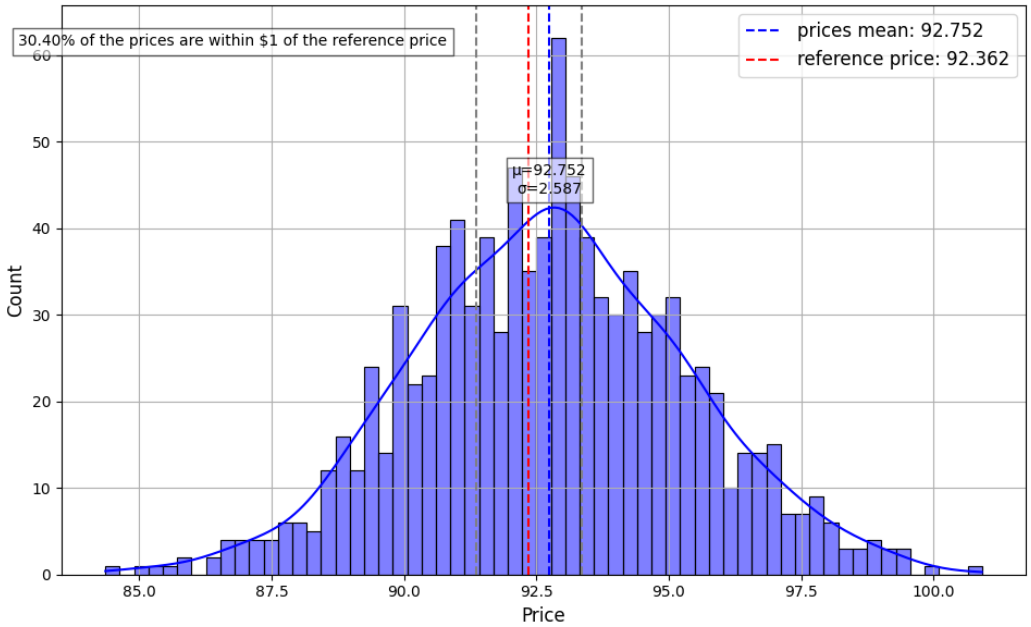
As introduced in section 5.5, when the initial wealth to be allocated in the hedging portfolio is not fixed a priori, the hedging solver implemented in this work provides an estimate, over the Monte Carlo replications of the hedging, of the initial wealth invested in the portfolio construction in order to guarantee an effective hedging. From various test conducted, most of the times this estimate proved to be close to the initial price of the target asset: this is reasonable, since the hedging solver recognizes that, to hedge a short position in the asset, a reliable value for the initial wealth should match the target asset's price quite accurately. Indeed, a lower value could result in under-hedging solutions, with potential losses, while a higher value could lead to over-hedging, accurately replicating the target payoff but with an excessive cost for the strategy, again resulting in potential losses.

In this section, a deeper analysis is provided by comparing the initial wealth required by the hedging strategy with the given price of the target asset. In particular, for target European options the reference price is computed by means of Black-Scholes-Merton formulas, while for Asian options it is estimated through a Monte Carlo simulation by averaging the discounted payoff over 10^4 replications. The key point of this test lies in not fixing the initial wealth of the hedging strategy and analyzing the distribution (over the set of MC replications) of the initial wealth that the solver exploits to construct the hedging portfolio. The number of MC replications considered is $n = 1000$ and the hedging strategy applied is self-financing and without the possibility to withdraw capital.

Figure 5.17a shows the distribution of the the initial wealth required by the solver to hedge against a European call option whose B&S price at the initial time of the hedging horizon is \$26.013. Similarly, Figure 5.17b shows the distribution of prices estimated via the hedging solver for an Asian put option with a reference price of \$92.362.
In the figures, the initial wealth of the various MC replications is denoted, for the sake of simplicity, by 'price'. The red vertical line indicates the reference price of the options (B&S price for the European target, Monte Carlo price for the Asian target), while the gray lines delimit a symmetric neighborhood of radius 1 around the reference price, to which belong 86.50% of the prices found by the hedging solver for the case of the European option and 30.40% of prices for the Asian option.



(a) European call



(b) Asian put

Figure 5.17: Distributions of the initial wealth required by the solver across multiple MC replications for hedging European and Asian options. The distributions are compared to the reference price of the target option.

If the mean of the initial wealth can be considered an estimator for the option (theoretical) price, this test confirmed that for European options the hedging solver returns a quite accurate estimate (the distribution mean is 26.68, close to the B&S price of $26.01, with a low standard deviation, that is 0.29).

Also for the Asian option, despite providing a less accurate estimator, the hedging solver returns a price distribution whose mean is close to the reference price (92.752 against a reference value of 92.362, with a standard deviation of 2.587, which is higher than the previous case but still acceptable).

To summarize, Table 5.12 reports the results discussed for the two tests conducted; in this table, the target option reference price is denoted by premium.

target asset		**price estimator**		**% of prices**
type	**premium**	**mean**	**std**	**within $1**
European call	26.01	26.68	0.29	86.50%
Asian put	92.362	92.752	2.587	30.40%

Table 5.12: Target option prices estimated through the hedging solver compared with the reference price.

Chapter 6

Benchmarking stochastic optimization

Following the performance analysis chapter, where the effectiveness of the hedging methodology via Stochastic Optimization was evaluated by deeply analyzing its sensitivity to different shades of the hedging configuration, this chapter presents a comparative analysis with two well-known approaches in the hedging literature to benchmark the proposed strategy. As previously mentioned, the Delta Hedging and Deep Hedging approaches will be considered.

6.1 Deep Hedging

As discussed in the introductory section 1.1.2, a Deep Hedging strategy known in the literature is used as a benchmark for the proposed strategy of dynamic hedging through stochastic optimization. In particular, the Deep Hedging strategy considered is the one implemented by A.Carbonneau in the paper *Deep hedging of long-term financial derivatives* [2]. In the following section, the main implementation aspects of the Deep Hedging approach are outlined, with a focus on the network architecture and its working mechanism.

6.1.1 Long-Short Term Memory networks

The network architecture chosen for the implementation of the Deep Hedging strategy is a Long-Short Term Memory Neural Network as the one implemented by Carbonneau. A LSTM is a specific type of Recurrent Neural Networks, i.e. a class of NNs which maps input sequences $\boldsymbol{x} = [x_1, \cdots, x_T]$ to output sequences $\boldsymbol{y} = [y_1, \cdots, y_T]$ and presents self-connections in the network layers; more precisely, RNN layers are functions not only of an input from the current time-step t but

also depend on their own output from the previous time-step $t-1$; this is the key feature of RNNs responsible for their ability to capture time dependence in input sequences. Figure 6.1 briefly shows the structure of a RNN with self-connections of layers; the input sequence is processed by the RNN timestep by timestep, that is, various processing steps follow each other, progressively taking as input the t-th component of the input sequence (that is, x_t) and outputs the t-th component of the output sequence (that is, y_t), for $t = 1, ..., T$.

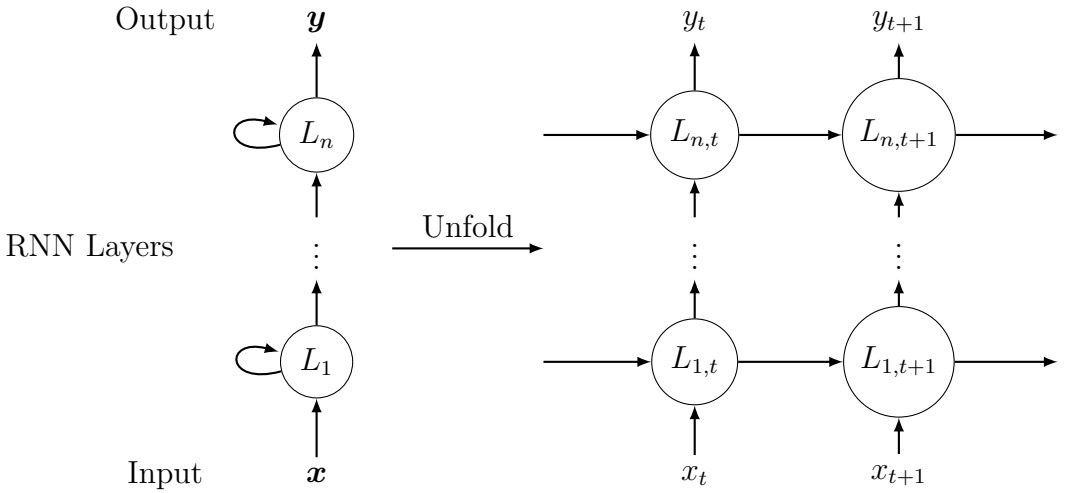


Figure 6.1: Structure of a recurrent neural network with self-connections of layers.

A LSTM network expands the idea of a RNN by precisely structuring the information stored from a time-step and used to influence the following time-step through the self-connection of layers (which, for the LSTM NN, are called *cells*). In particular, this type of networks feature a hidden state h_t and a cell state c_t; the former denotes a short-term memory and represents the output of the cell at time-step t, which is passed to the following cell, if there is one, in the sequence of layers and it is also passed to the cell itself through a self-connection. The latter, instead, is responsible for storing the long-term memory of the past time-steps: indeed, c_t contains (part of) the past information of all the previous times-steps. The information storage in the cell state happens through a set of gates, specifically the input, the output and the forget gates (i_t, o_t, f_t), which are responsible for choosing the information to store and the one to forget. Going deeper into details:

- the input gate i_t decides how much information from the current time-step should be added to the cell state c_t;
- the forget gate f_t decides how much information from the previous step (represented by c_{t-1}) should be discarded when generating the cell state c_t;

- the output gate o_t decides how much of the cell state information is passed to the hidden state h_t.

Denoting by $j \in J = \{1, \dots, H\}$ the index for cells in the sequence of the network layers and with $t \in \{1, \dots, T\}$ the index for time-steps, the structure of the j^{th} LSTM cell is represented by Figure 6.2.

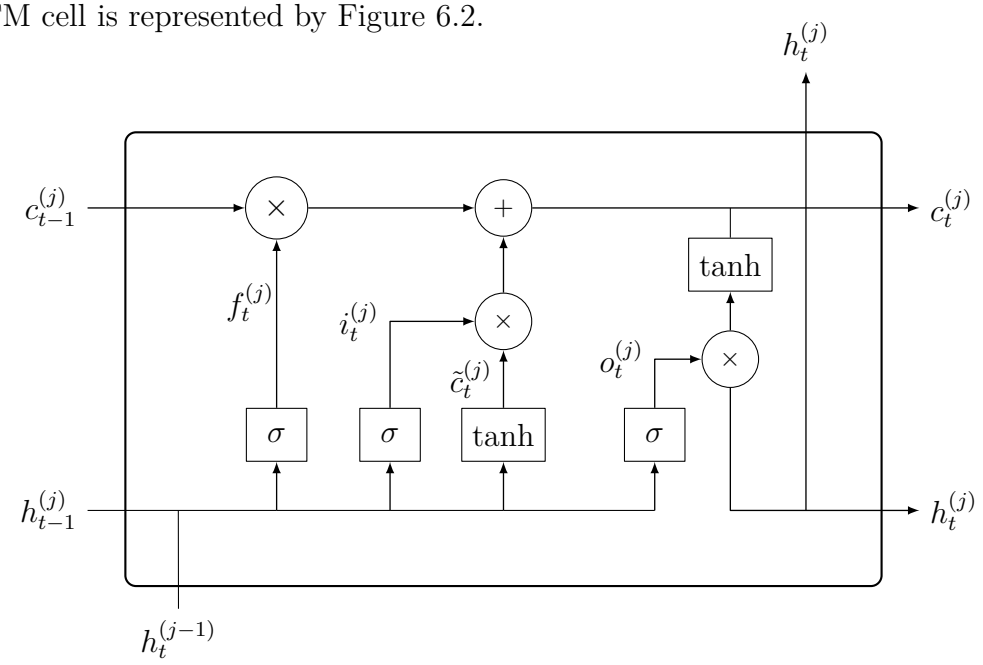


Figure 6.2: General structure of a LSTM cell.

For the sake of clarity, note that:

- inputs $h_{t-1}^{(j)}$ and $c_{t-1}^{(j)}$ for a cell correspond to the outputs of the cell itself at the previous time-step; thus, this values are passed from time-step $t-1$ to time-step t by means of the self-connection of the cell;
- for the first cell (that is, $j = 1$), it holds that the input $h_t^{(j-1)}$ is the input of the network at time-step t, that is x_t;
- the output $h_t^{(H)}$ of the last cell is processed through an additional layer, known as the *output layer*, where a linear transformation is applied to $h_t^{(H)}$ by means of a weight matrix W_y and a bias term b_y to compute the output of the network at time-step t, that is y_t:

$$y_t = W_y \, h_t^{(H)} + b_y;$$

- for every other cell (that is, $j = 2, \dots, H-1$), the input $h_t^{(j-1)}$ corresponds to the output of the previous cell at the current time-step t and the output $h_t^{(j)}$ is passed to the following cell.

To further clarify, the formulas of the LSTM network components are provided below:

$$i_t^{(j)} = \sigma\Big(W_i^{(j)}\Big[h_{t-1}^{(j)},\, h_t^{(j-1)}\Big] + b_i^{(j)}\Big), \tag{6.1}$$

$$f_t^{(j)} = \sigma\Big(W_f^{(j)}\Big[h_{t-1}^{(j)},\, h_t^{(j-1)}\Big] + b_f^{(j)}\Big), \tag{6.2}$$

$$o_t^{(j)} = \sigma\Big(W_o^{(j)}\Big[h_{t-1}^{(j)},\, h_t^{(j-1)}\Big] + b_o^{(j)}\Big), \tag{6.3}$$

$$c_t^{(j)} = f_t^{(j)} \odot c_{t-1}^{(j)} + i_t^{(j)} \odot \tanh\Big(W_c^{(j)}\Big[h_{t-1}^{(j)},\, h_t^{(j-1)}\Big] + b_c^{(j)}\Big), \tag{6.4}$$

$$h_t^{(j)} = o_t^{(j)} \odot \tanh\Big(c_t^{(j)}\Big), \tag{6.5}$$

where:

- $\sigma(\cdot)$ and $\tanh(\cdot)$ are respectively the sigmoid and hyperbolic activation functions;
- $h_t^{(0)}$ corresponds to the input vector x_t;
- $h_t^{(H)}$ corresponds to the output of the layers, through which y_t is computed by the linear transformation mentioned before;
- the parameters in the set $\{W_i^{(j)}, W_f^{(j)}, W_o^{(j)}, W_c^{(j)}\} \cup \{b_i^{(j)}, b_f^{(j)}, b_o^{(j)}, b_c^{(j)}\}$, along with the previously mentioned W_y and b_y, denote the cell weights and biases; these are the parameters whose optimal value is aimed to learn through the network training phase (see subsection 6.1.2).
- $\odot$ denotes the scalar product.

6.1.2 Implementation

Inspired by the work of A.Carbonneau, in this section the implementation of the Deep Hedging approach as a benchmark for the hedging strategy based on Stochastic Optimization will be examined.

Similarly to the SO hedging framework, the market is in discrete time and the hedging strategy consists of two phases: at first, the hedging portfolio is constructed, then, at subsequent stages, the latter is rebalanced. The hedging strategy is denoted by the sequence $\boldsymbol{\delta} = \{\boldsymbol{\delta}_t\}_{t=1}^T$, where each component $\boldsymbol{\delta}_t$ represents the vector of holdings in the hedging assets $j \in [1, \ldots, n]$ held in the time interval $[t-1, t]$:

$$\boldsymbol{\delta}_t = [\delta_t^{(1)}, \ldots, \delta_t^{(n)}].$$

The LSTM network plays the role of approximating the hedging strategy: indeed, for each input sequence $\boldsymbol{x} = \{\boldsymbol{x}_t\}_{t=1}^T$, the network outputs the sequence $\boldsymbol{\delta} = \{\boldsymbol{\delta}_t\}_{t=1}^T$. Each component $\boldsymbol{x}_t$ of the input sequence is a vector containing the prices of the assets (following the denotation under which $x_t^{(j)}$ denotes the price of the j^{th} asset at time t) and the value of the hedging portfolio at time t (normalized by its initial value), that is:

$$\boldsymbol{x}_t = \left[x_t^{(0)}, \ldots, x_t^{(n)}, \frac{V_t^h}{V_0^h} \right];$$

the last component is included into $\boldsymbol{x}_t$ as it is shown by [2] that this additional information enhances the training process of the network.
The chosen configuration for the LSTM network features $H = 2$ cells (layers), for a total of three layers if the output layer is included (note that the output layer corresponds to the linear transformation applied to $h_t^{(H)}$ for the computation of $\boldsymbol{\delta}_t$).

Since the Deep Hedging method is data-driven, the generation of a training, validation and test set is required[1]; these sets of simulated paths for the financial market are generated in the same way as for the SO hedging; in particular, the simulated dataset includes 50000 (MC) replications of the market evolution. The model is trained over 35000 of those simulated paths, while additional 7500 are used as a validation set; the remaining 7500 out-of-sample paths are used to test the performance of the Deep Hedging algorithm. During the training phase, the network learns the optimal parameters by minimizing a risk measure represented by a loss function $L(\cdot)$ applied to the hedging error; recalling that the latter is the difference between the target asset payoff at maturity Φ_T and the hedging portfolio value at the same date $V_T^h(\boldsymbol{\delta})$, the Deep Hedging objective is:

$$\min_{\boldsymbol{\delta}} \mathbb{E}\left[L(\Phi_T - V_T^h(\boldsymbol{\delta})) \right].$$

The chosen loss function is the Semi-Mean Square Error (SMSE), defined as

$$L(x) = x^2 \mathbf{1}_{\{x>0\}},$$

as shown by Carbonneau to lead to better results compared to its symmetric counterpart (the MSE function).

During the training phase, the network parameters θ (including weights and

[1]This structure is typical of data-driven methods, in which the optimal model parameters are learned through the data in the training set, while the validation set is used to select the best choices for the model hyper-parameters. After the training procedure, data in the test set are used to test the optimized model configuration and to provide performance statistics.

biases of every LSTM cell, as well as those of the output layer) are optimized by means of the *stochastic gradient descent* (SGD) algorithm. The latter is an optimization algorithm inspired by the Gradient Descent algorithm, despite updating the parameters according to a random subset of the data, rather than exploiting the entire dataset. Denoting by $J(\theta)$ the objective function to minimize, that is

$$J(\theta) = \mathbb{E}\left[L(\Phi_T - V_T^h(\boldsymbol{\delta_\theta}))\right],$$

the (well-known) Gradient Descent algorithm would update iteratively the network parameters through the formula given by:

$$\theta_{j+1} = \theta_j + \eta_j \nabla J(\theta_j),$$

where η_j is known as *learning rate*; the idea under the SDG is to approximate, at each training step j, the gradient of the objective function by an estimate $\nabla \hat{J}(\theta_j)$ computed over a random subset B (denoted as *batch*) with cardinality N of the available data:

$$\theta_{j+1} = \theta_j + \eta_j \nabla \hat{J}(\theta_j), \tag{6.6}$$

where

$$\hat{J}(\theta_j) = \frac{1}{N} \sum_{i \in B} L\left(\Phi_{T,i} - V_{T,i}^h(\boldsymbol{\delta_{\theta_j}})\right).$$

The chosen batch size for the implementation of the LSTM in this work is $N = 500$, with 150 training steps (formally denoted as *epochs*) in which the SGD updating rule of Equation 6.6 is applied. For the specific implementation considered, the validation set is not exploited to choose the best configuration of the model hyperparameters (e.g. number of cells, number of neurons per cell and more) but rather to select the optimal configuration of the model parameters θ^* out of the 150 training epochs[2].

Without entering into specific details, the LSTM is trained by combining SGD with the *Adam optimizer* and the *Backpropagation Through Time* (BPTT) algorithm. The former is responsible for updating the learning rate η_j between the training steps j and $j+1$, leveraging the first and second moments of the gradient; the adaptive learning rates of the Adam optimizer result in a faster convergence of the training phase. The latter, instead, is an alternative to the classic backpropagation method commonly used in applications involving recurrent neural networks.

[2]This choice is motivated by Carbonneau by the fact that different choices for the model hyper-parameters have minor impact on the hedging performance.

6.1.3 Hedging setup

To benchmark the proposed hedging strategy via Stochastic Optimization with the Deep Hedging approach, the following configuration is considered:

- the target asset to be hedged is an Asian put option with the following features:
  - underlying stocks in the set {ENEL.MI, MMM, TSLA} with initial prices [102.05, 242.92,107.19],
  - strike price $K = 250$,
  - maturity $T = 1$ year;
- hedging assets including the underlying stocks, the risk-free asset and one European put option for each available stock, with strike prices [170,300,200], respectively for the mentioned stocks, and all with maturity 1 year;
- the number of Monte Carlo replications to test the strategy is $n = 7500$. The Deep Hedging strategy, as mentioned above, is trained over 35000 simulated paths, with additional 7500 samples for the validation set;
- both hedging strategies have been implemented according to their best performing setups (trading-off the hedging performance and the computational time required):
  - the SO hedging strategy involves 3 decision stages with scenario trees generated at each stage according to the branching factors [25,3,3], with backward erosion approach;
  - the Deep Hedging strategy is implemented with monthly decision stages, incurring thus in 12 decision stages.

6.1.4 Results

Figure 6.3 and Figure 6.4 show the empirical Profit&Loss distributions for the training and test sets, respectively, of the Deep Hedging strategy, while Figure 6.5 presents the same for the strategy proposed in this work. Before discussing the comparative analysis of the two different strategies, one aspect of the Deep Hedging performance deserves particular attention. Indeed, as shown in Table 6.1, which summarizes the statistics obtained for the P&L of both the DH training and test sets, it turns out that the hedging performance is very similar for these two sets: this highlights a minimal impact of the generalization error and demonstrates the absence of overfitting during the training phase.
Having observed this behavior of the Deep Hedging, the focus now shifts to comparing the results of the two different methodologies. For this analysis, only the test

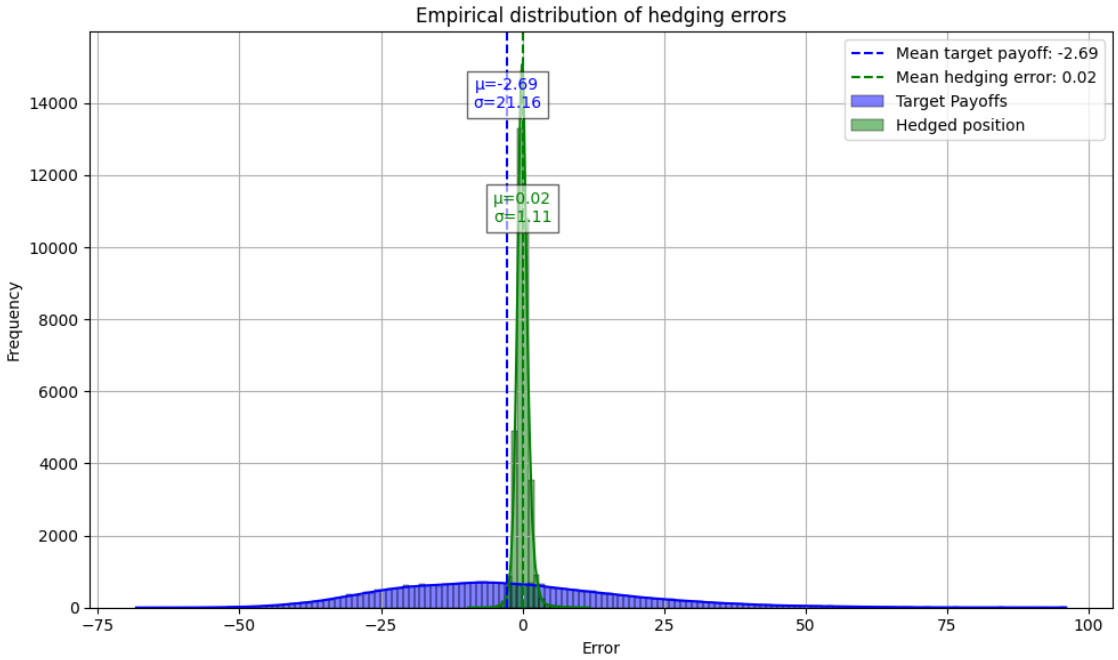


Figure 6.3: Deep Hedging - PnL distribution of the training set.

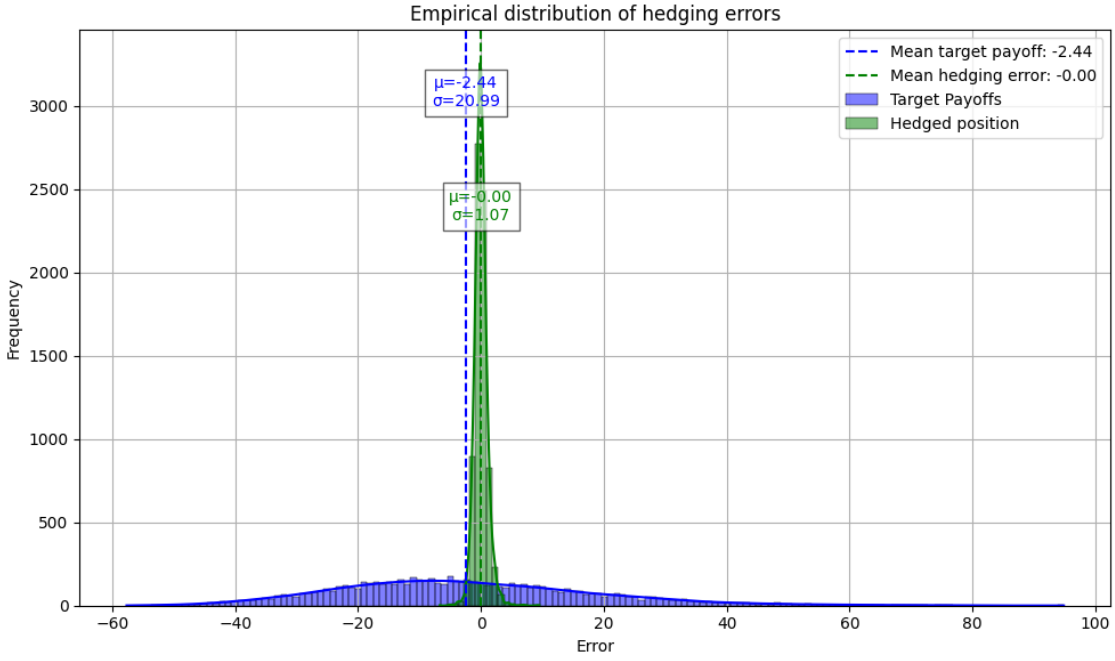


Figure 6.4: Deep Hedging - PnL distribution of the test set.

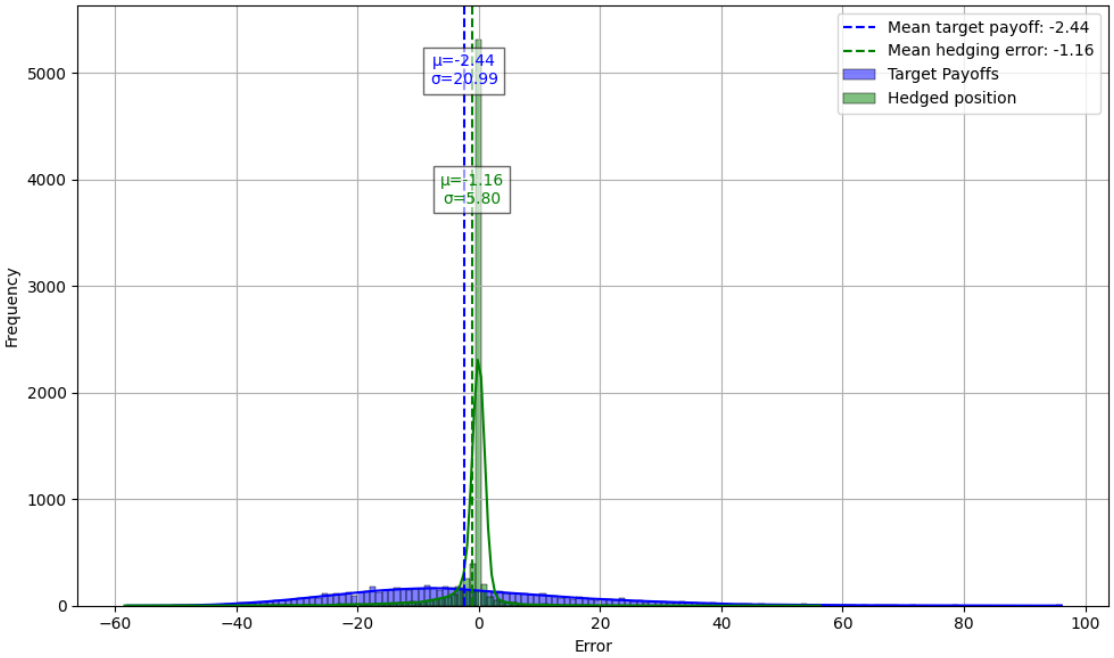


Figure 6.5: Performance of the Stochastic Optimization hedging in the comparative analysis with the Deep Hedging approach.

	Hedged P&L		**Unhedged P&L**	
set	mean	std	mean	std
Training	0.02	1.11	−2.69	21.16
Test	0.00	1.07	−2.44	20.99

Table 6.1: Statistics of the performance of the Deep Hedging training and test sets.

set of the Deep Hedging approach is considered, willing to analyze the performance of the DH strategy in new scenarios that were not involved in the network training phase.

Based on a first observation of the provided figures, the conclusion would be clear: both Deep Hedging and Stochastic Optimization hedging successfully hedged against the liability related to the target payoff, decreasing the standard deviation of the unhedged position and with a hedged P&L which is quite accurately centered at zero. Going deeper into the analysis, a discussion can be made around some

discrepancies in the method's behavior. Indeed, the difference in the mean values for the P&L distributions is negligible (0 for the DH approach against -1.16 for the SO hedging, both reasonable values), but the DH approach proves to have a higher stability in the hedging strategy, with a lower value for the standard deviation of the P&L distribution with respect the the SO strategy (1.07 against 5.80); this reflects the heavier upper and lower tails in the SO distribution. Indeed, the following are true:

- the SO approach leads to a distribution that counts the highest number of replications where the P&L (in this case equal to the hedging error) is exactly zero or a value close enough to zero to be approximated as such, indicating a perfect hedged situation; specifically, out of the 7500 MC replications, more than 5000 have a P&L with value zero (or close). Despite that, the tails of its distribution reveal that some worse cases, where the P&L deviates from zero, may occur, although they remain isolated cases of lower frequency and with P&L values still acceptable;

- on the other hand, the DH distribution counts a lower number of replications exactly within a small neighborhood of the value zero (specifically, around 2750 out of the 7500 MC replications), but its standard deviation remains low due to the fact that even those values that are not exactly zero still lie around it, with a lower number of outlier behaviors: this reflects in thinner tails for the distribution.

Concerning computational time requirements, the Deep Hedging strategy turns out to be more time-consuming, with a total time for the network optimization phase (including both training and validation) of approximately 2 hours and 37 minutes; once the optimal network parameters are learned, the test phase is quite immediate, adding no significant burden to the total amount of time required. On the other hand, the Stochastic Optimization strategy is somewhat less time-consuming, with a total requirement of 1 hour and 49 minutes to apply the hedging strategy to the set of test scenarios. Both strategies are not immediate and necessitate for an amount of time that is not negligible, but SO hedging demonstrates a better time efficiency; despite that, the higher computational cost of the DH strategy may be acceptable given the (slight) improvement in P&L performance.

Table 6.2 summarizes the statistics computed for this comparative analysis between the Deep Hedging and the Stochastic Optimization hedging approaches. In the table compares an additional risk measure for the P&L distribution, that is the Conditional Value at Risk ($\mathrm{CV@R}_\alpha(\mathcal{L})$), which is a defined as:

$$\mathrm{CV@R}_\alpha(\mathcal{L}) = \mathbb{E}\left[\mathcal{L} | \mathcal{L} < \mathrm{V@R}_\alpha(\mathcal{L})\right]$$

method	**P&L**			**time**
	mean	std	$\text{CV@R}_{0.05}$	
Stoch. Opt. Hedging	-1.16	5.80	-20.25	1h 49min
Deep Hedging	0.00	1.07	-2.78	2h 37min

Table 6.2: Summary of statistics from the comparative analysis between the Deep hedging approach and the hedging strategy implemented through Stochastic Optimization.

where V@R_α is defined as:

$$\begin{aligned}\text{V@R}_\alpha(\mathcal{L}) &= \inf\{x \in \mathbb{R} : \mathbb{P}(\mathcal{L} \leq x) \geq \alpha\} \\ &= \inf\{x \in \mathbb{R} : \mathbb{F}_{\mathcal{L}}(x) \geq \alpha\}.\end{aligned}$$

To better clarify, if the distribution of $\mathcal{L}$ is continuous, then V@R_α would be the quantile of level α.

6.2 Delta Hedging

The second benchmark assessed for the hedging strategy proposed in this work is Delta Hedging, briefly introduced in section 1.1.2. For the sake of simplicity, only target asset with a single underlying are considered for the Delta hedging implementation[3]. Assuming a short position in a target option and recalling the Delta is defined as the rate of variation of the option price with respect to variations in the underlying price, the Delta Hedging aims at maintaining a delta-neutral portfolio in order to minimize the risk exposure due to changes in the underlying price, preserving the initial portfolio value. This is accomplished by trading the underlying at each rebalancing date in order to hold an amount of the latter equal to the delta of the option.

The Delta Hedging strategy implemented is the one described by M.Villaverde in [1]. In contrast to what was introduced in section 1.1.2, the Delta Hedging strategy may be applied in an asset-trading framework that accounts for transaction costs. Denoting by:

- B_t the value of the risk-free asset at time t,

[3]For more complex options, there may not be an analytical derivation of the Delta Hedging strategy, but also in cases where it exists, its implementation may not be trivial.

- S_t the price of the underlying stock at time t,
- $X_t = (X_t^0,\ X_t^1)$ the vector of assets' holdings in the portfolio at time t, where X_t^0 refers to the risk-free asset and X_t^1 to the stock and
- W_0 the initial wealth invested in the portfolio construction,

according to Villaverde's strategy, if a proportional transaction cost c for the stock is taken into account, the holdings of the assets in the hedging portfolio should be:

$$X_0^1 = \Delta_0, \quad X_0^0 = \frac{W_0 - X_0^1 S_0(1+c)}{B_0},$$

$$X_t^1 = \Delta_t, \quad X_t^0 = X_{t-1}^0 + \frac{(X_{t-1}^1 - X_t^1)S_t - |X_t^1 - X_{t-1}^1|S_t c}{B_t} \quad \forall\ t = 1, \ldots, T-1$$

and

$$X_T^1 = \Delta_{T-1}, \quad X_T^0 = X_{T-1}^0$$

The comparative framework implemented for this benchmark involves hedging a European put option with strike $K = 300$, maturity $T = 1$ year and underlying stock corresponding to the ticker TSLA with a self-financing strategy. Transaction costs are set different from zero for both strategies in order to assess the results in a more general framework, even if the Delta Hedging is derived under the B&S assumptions, which require no market frictions. One key aspect of the implementation concerns the use of branching factor 2 for each rebalancing stage of the SO hedging, that is, the branching factors typical of the binomial model.

Analysis on a high number of decision stages

An initial comparative analysis focuses on the implementation of both strategies for a dynamic hedging involving several decision stages. In particular, 9 rebalancing stages are considered, with an initial scenario tree for the Stochastic optimization approach which features 2^9 simulated scenarios. Figure 6.6 shows the P&L for the SO hedging, while the Delta Hedging P&L is reported in Figure 6.7.
Delta Hedging exhibits a smaller standard deviation, suggesting a (slightly) more stable outcome, but in general the results obtained for this first analysis are promising for both strategies, highlighting their effectiveness.
In terms of computational time required, there is a significant difference when comparing stochastic optimization and Delta Hedging: the computational time for the former amounts to 2219.05 s for the simulation tested, while the latter requires only 0.003 s. This discrepancy is motivated by the two methods complexity: Stochastic optimization is an advanced technique for which various optimization problems are solved at each decision stage and relies on advanced techniques

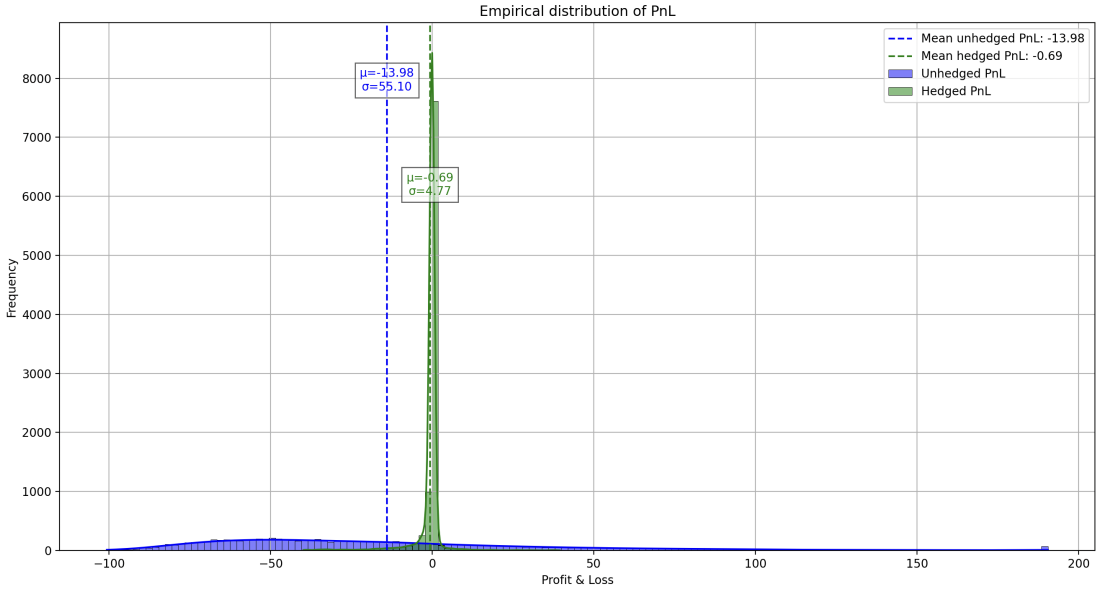


Figure 6.6: Profit and Loss of the stochastic optimization hedging for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 9 decision stages.

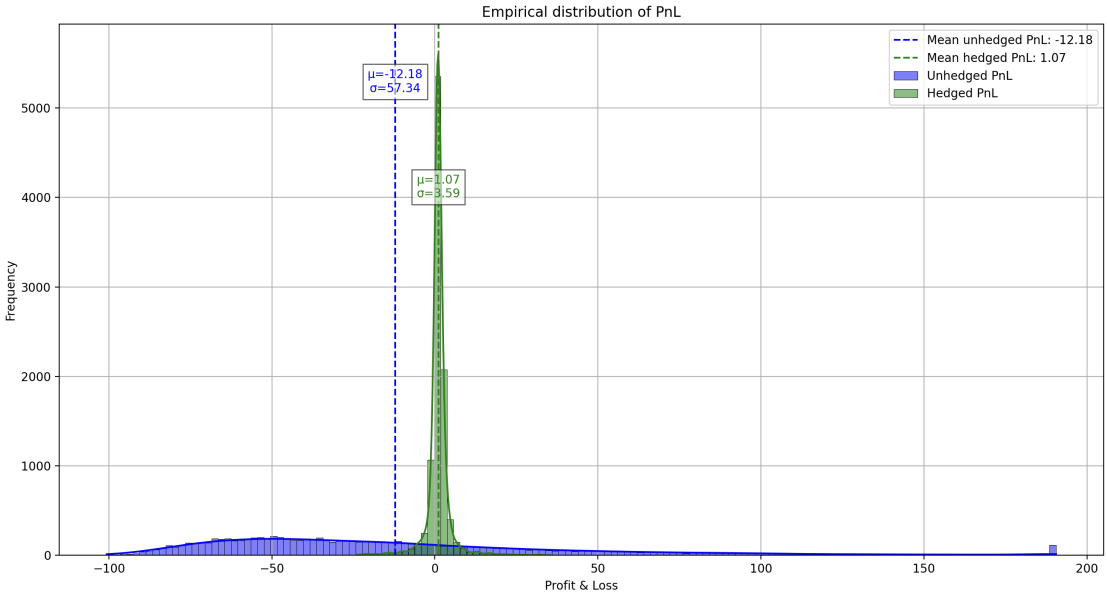


Figure 6.7: Profit and Loss of the Delta Hedging of a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 9 decision stages.

(among the others, scenario trees to model uncertainty), naturally increasing the computational burden. Delta Hedging, instead, is a straightforward strategy that simply adjusts portfolio holdings at each decision stage based on current values of the hedging assets, following deterministic formulas. In light of this, Delta Hedging requires minimal computational effort.
However, another aspect to take into account is the methods adaptability, for which the stochastic optimization approach outperforms Delta Hedging: indeed, the latter relies on a specific pricing model for the target option to be hedged based on strict assumptions; on the other hand, the SO hedging is flexible to various hedging settings, leaving the door open to a variety of assumptions about the financial market and its dynamics.

Analysis on a fewer number of decision stages

The previous test could potentially advantage Delta Hedging due to the many rebalancing steps: in fact, for a higher choice of branching factors, the SO approach could have required an excessive computational time considering the 9 decision stages, while the choice of branching factors equal to 2 could lead to a not-so-precise approximation of possible future outcomes. A further test is conducted willing to analyze the behavior of the two strategies with a different choice for the number of decision stages (and branching factors), that is, 3 stages (with branching factors for the SO hedging represented by the vector [25,3,3]). Figure 6.8 and Figure 6.9 show the P&L empirical distributions of the two methods under the mentioned conditions. Even with a smaller number of rebalancing stages, both strategies provide a good quality solution in terms of P&L distribution, with a slightly better performance for the stochastic optimization approach, which features both the mean and standard deviation very close to zero.

A further consideration to keep in mind is that Delta Hedging formulas derive from a first-order approximation, making the delta an effective method to hedge only against small price movements in the underlying asset. For stress tests in which the underlying price features higher variability and may also encounters jumps, Delta Hedging could potentially fail to hedge the position in the target asset, while the stochastic optimization approach would be more robust due to its ability to model and take into consideration possible future outcomes.

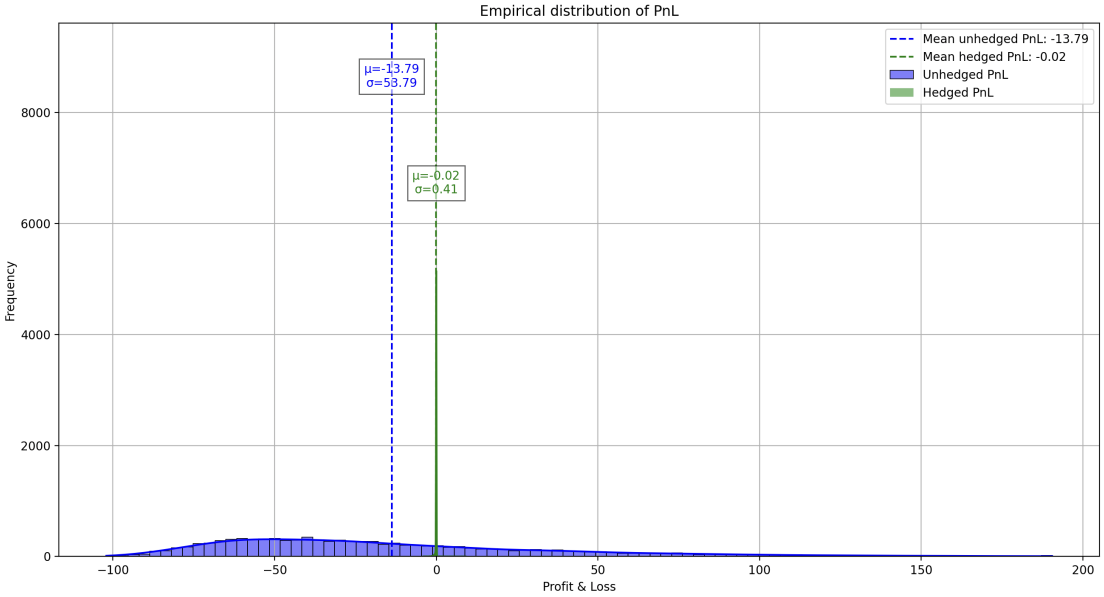


Figure 6.8: Profit and Loss resulting from stochastic optimization for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 4 decision stages.

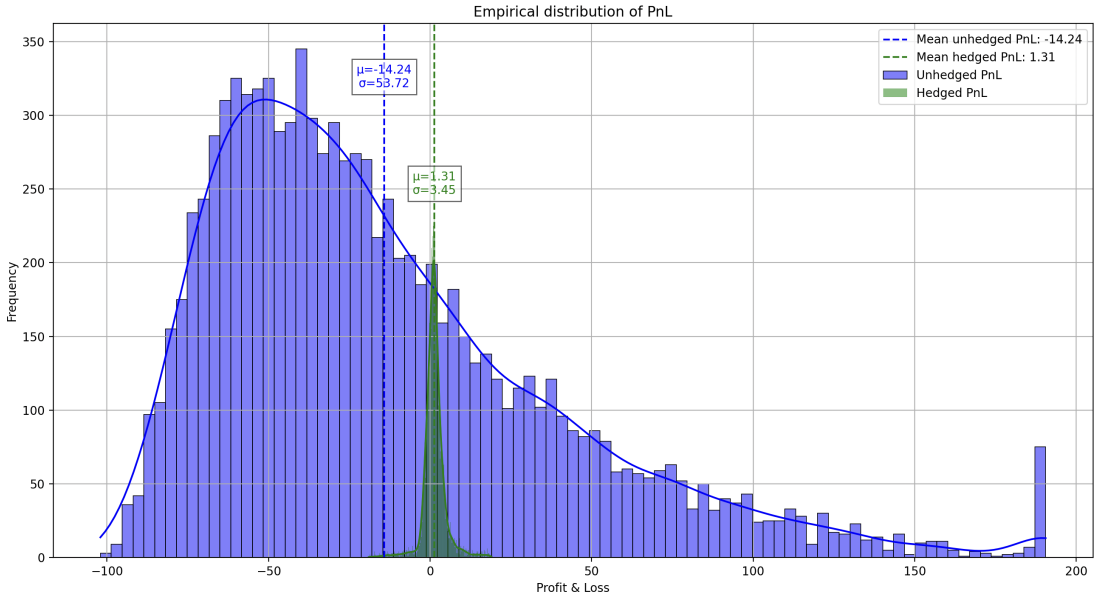


Figure 6.9: Profit and Loss resulting from Delta Hedging for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 4 decision stages.

Chapter 7

Conclusion

This thesis has presented and developed a Stochastic Optimization approach for the dynamic hedging of exotic derivatives. The study began with an introduction to the topic, highlighting the motivation behind a hedging strategy and its objective of risk management. The theory of financial markets provided the foundation to establish the framework of the proposed hedging strategy: the financial instruments involved in the hedging process were defined along with their characteristics and dynamics. The assumed stochastic process governing stock prices evolution is the Geometric Brownian Motion, implying a log-normal distribution for stock prices. Derived from this assumption on stock dynamics, the Black-Scholes-Merton model was employed for pricing European options, while for exotic derivatives, such as Asian options, the price was estimated through Monte Carlo simulations. The dynamic hedging strategy was then examined: a detailed description of the implemented approach was given, analyzing the stochastic optimization problem solved at each decision stage to find the optimal portfolio composition, along with the generation of scenario trees, leveraged to tackle the issue related to the uncertainty about possible future outcomes in the financial market. Different approaches to scenario tree generation were proposed, involving the simulation of prices evolution through the Geometric Brownian Motion and the simulation through the moment matching method: the results showed that all the proposed strategies to scenario tree generation resulted in good quality results, confirming their effectiveness. Following the definition of the optimization problem, some performance analyses were conducted leveraging Monte Carlo simulation to assess the effectiveness of the hedging strategy in minimizing the risk exposure related to the short position in the target asset. Results obtained showed that the proposed approach leads to satisfactory performances, highlighting its effectiveness. Key findings from this testing phase include that the self-financing strategy with fixed initial wealth equal to the target asset premium proved to be the optimal hedging configuration, transaction costs demonstrated to have not a significant impact on

the hedging performance, pointing out the robustness of the proposed strategy across various scenarios simulating real financial markets, and, additionally, that the hedging strategy may also be leveraged to provide a reliable estimate of the price of exotic derivatives. In the final phase, to benchmark the hedging strategy developed in this thesis, the Deep Hedging and Delta Hedging approaches were considered. The results from this comparative analysis showed that the hedging approach via stochastic optimization performs well among the benchmark alternatives, with each strategy presenting its own strengths and weaknesses, but overall exhibiting comparable results.

Bibliography

[1] Michael Villaverde. «Hedging European and Barrier options using stochastic optimization». In: *Quantitative Finance* 4.5 (2004), pp. 549–557 (cit. on pp. 4, 18, 28, 34, 93).

[2] Alexandre Carbonneau. *Deep Hedging of Long-Term Financial Derivatives*. `https://github.com/alexandrecarbonneau/Deep-Hedging-of-Long-Term-Financial-Derivatives`. Accessed: 2024-08-26. 2020 (cit. on pp. 6, 83, 87).

[3] Yuliya Romanyuk. *Asset-Liability Management: An Overview*. Discussion Paper 2010-10. Bank of Canada, Aug. 2010. URL: `https://www.bankofcanada.ca/2010/08/discussion-paper-2010-10/` (cit. on p. 31).

[4] John C. Hull. *Options, Futures, and Other Derivatives*. 9th. Prentice Hall, 2017.

[5] Marek Capinski and Tomasz Zastawniak. *Mathematics for Finance: An Introduction*. Springer, 2003.

[6] Jacek Gondzio, Roy Kouwenberg, and Ton Vorst. «Hedging options under transaction costs and stochastic volatility». In: *Quantitative Finance* 2.4 (2002), pp. 310–321.

[7] Alexandre Carbonneau. *Equal-risk Pricing with Deep Hedging*. `https://github.com/alexandrecarbonneau/Equal-risk-pricing-with-deep-hedging`. Accessed: 2024-08-26. 2020.

[8] Teemu Pennanen and Antero Ranne. «A stochastic programming model for asset-liability management». In: *Annals of Operations Research* (2007). DOI: `10.1007/s10479-006-0135-3`.