POLITECNICO DI TORINO

Master's Degree in Mathematical Engineering

Master's Degree Thesis

Hedging exotic derivatives via stochastic optimization models: a focus on Asian and Barrier Options and Worst Performance derivatives

Supervisors

Prof. Paolo BRANDIMARTE

Prof. Edoardo FADDA

Ph.D. Giovanni AMICI

Candidate

Ioana REUT

March 2025

Summary

Stochastic optimization techniques are applied to the framework of hedging exotic options from the perspective of a bank which sells derivatives and thus is exposed to potential future liabilities. The objective is to formulate an optimal strategy that minimizes the impact of potential losses. In practice, hedging involves the construction of a portfolio, referred to as hedging portfolio, which will be periodically adjusted in response to market changes that have an impact on the considered derivative. The optimization process relies on scenario trees generated through stochastic models. Two methods for simulating underlying stock prices (Geometric Brownian Motion and Moment Matching) are presented. Several optimization problems are then developed and compared based on their hedging performance, associated costs and profit and loss. Self-financing and non-self-financing strategies are analyzed and compared, exploring also rebalancing frequencies, transaction costs and their influence on overall hedging performance. Additionally, the scenario trees structure is carefully analyzed to explore the trade-off between accuracy of the results and computational time. The analysis focuses on covering exotic derivatives, such as Asian and Barrier options and Worst Performance derivatives, whose structural features and valuation models are explained in detail. European vanilla options are also used and considered as benchmarks to validate the models and compare stochastic optimization with traditional delta hedging.

Table of Contents

List of Tables

List of Figures

Acronyms

WP
: Worst-Performance

WP1
: Standard Long Barrier Plus Worst of Certificates

WP2
: Standard Long Barrier Digital Worst of Certificates

WP3
: Standard Long Autocallable Barrier Digital Worst of Certificates with memory effect

FVD
: Final Valuation Date

GBM
: Geometric Brownian Motion

MM
: Moment Matching

P&L
: Profit and Loss

ER
: Early Redemption

Chapter 1

Introduction

In financial markets, the trading and risk management of derivatives represent a challenge for financial institutions. In order to mitigate the risks associated with these products, hedging strategies can be employed. In this context, this thesis examines the problem of hedging exotic derivatives through stochastic optimization models, adopting the point of view of a bank that writes and sells financial instruments, thus being exposed to potential future losses. Among these, Asian options, Barrier options and Worst Performance derivatives stand out due to their non-trivial structures and sensitivity to market movements.
The problem is addressed within the Asset-Liability Management context: the bank has to fulfill a set of liabilities; to ensure sufficient funds, it allocates its available wealth in a diversified set of assets, whose future prices are represented by random variables since they are uncertain. The uncertainty in the model is represented by a discrete set of sample paths of stochastic processes for each random variable: here, the uncertainty in the problem results from the uncertainty in stock prices.
Stochastic optimization is a valid approach for dealing with financial decision-making problems under uncertainty. In the context of hedging exotic derivatives, stochastic optimization methods allow the bank to determine rebalancing strategies with the aim of finding a trade-off between the minimization of risk exposure and the cost of the adopted hedging strategy.
Stochastic optimization provides a versatile framework where various hedging error metrics can be considered (reflecting different risk aversions) and market frictions, such as transaction costs, can be easily taken into account. Additionally, since it is based on simulations, it can work with almost any stochastic model for stock prices.
The process of generating discrete outcomes for random variables is known as scenario generation. In this context, scenarios are distinct paths for the prices of the hedging assets and potential liabilities at each rebalancing time, from the current time to the end of the hedging horizon. The quality of price scenarios

plays a crucial role in determining the effectiveness of a hedging strategy designed through stochastic optimization.
More in detail, stochastic programming is performed through scenario trees, where each node represents a state of the world at a specific time.
The hedging optimization problem is a multi-stage stochastic problem: at the current time, the first decision, *here-and-now*, is made. Then, after observing the realization of the risk factor, a new decision is taken based on the updated information: in such stochastic problems, a sample path of the relevant risk factors is observed and then decisions are adapted along the day. This pattern of alternating decisions and observations continues throughout the hedging horizon. At each node of the scenario tree, the stochastic programming approach selects the optimal decision relying only on the available information at that time.
Since the market evolves in time, the model can be updated by observing the new market conditions. Thus, a new problem, based on new market conditions, is solved at each time step. This is what is called dynamic hedging: instead of making a decision at the current time keeping it unchanged until maturity, the decision is frequently adjusted through rebalancing to reflect market changes.

Stochastic optimization is therefore applied in this thesis to determine a suitable hedging strategy for the considered derivatives, allowing banks to activate them to prevent significant losses. The various hedging strategies that can be implemented in the framework of stochastic optimization are described in Chapter 2, along with a more classical approach, delta hedging, used as a benchmark for stochastic optimization formulations. The stochastic models assumed for simulating price paths are presented in Chapter 3, with a strong emphasis on the simulations of scenarios that are free of arbitrage opportunities. A discussion of the derivatives considered both as hedging instruments and as instruments to be hedged is provided in Chapter 4: a detailed discussion is dedicated to the structure and valuation methods of vanilla, Asian options and barrier options and worst-performance derivatives. Since the results supporting this work were obtained through a Python code implementation, Chapter 5 presents its most relevant aspects. Finally, Chapter 6 summarizes the key findings of the conducted analysis, also comparing stochastic optimization with the traditional delta hedging approach.

Chapter 2

Hedging

When a financial institution sells a derivative to a client, it has to address the issue of risk management: investing in a risky asset may result in periods of losses or returns that fail to meet expectations. Through *hedging*, it is possible to limit the exposure to risk and the impact of price fluctuation.
An ideal hedging (*perfect hedging*) would eliminate all risks entirely, but in reality such perfection is rarely achievable. However, several hedging strategies can be developed to minimize risk as effectively as possible, rather than completely eliminate it. The key approach consists in adjusting the exposure by taking positions in the underlying assets and, if needed, in related derivatives. In practice, hedging involves the construction of a portfolio, typically referred to as *hedging portfolio*, which will be periodically adjusted in response to market changes that have an impact on the considered derivative.
In a real-world setting where transaction costs exist, frequent rebalancing is not practical: a trade-off between risk mitigation and cost efficiency should be found. The following section explores dynamic hedging techniques using stochastic optimization: the bank aims to hedge the derivative over multiple future scenarios by planning portfolio rebalancing at specific rebalancing dates. The hedging instruments can include m underlying stocks, vanilla options and a money bank account (cash), each identified by an index $j \in \mathcal{A}$.

2.1 Main hedging problem: symmetric and asymmetric cases

The main optimization problem considered in determining the portfolio rebalancing strategy focuses on minimizing the deviation between the hedging portfolio value at maturity and the actual cash flow (or payoff) at maturity resulting from holding a short position in the derivative to be hedged. This quantity is referred to as

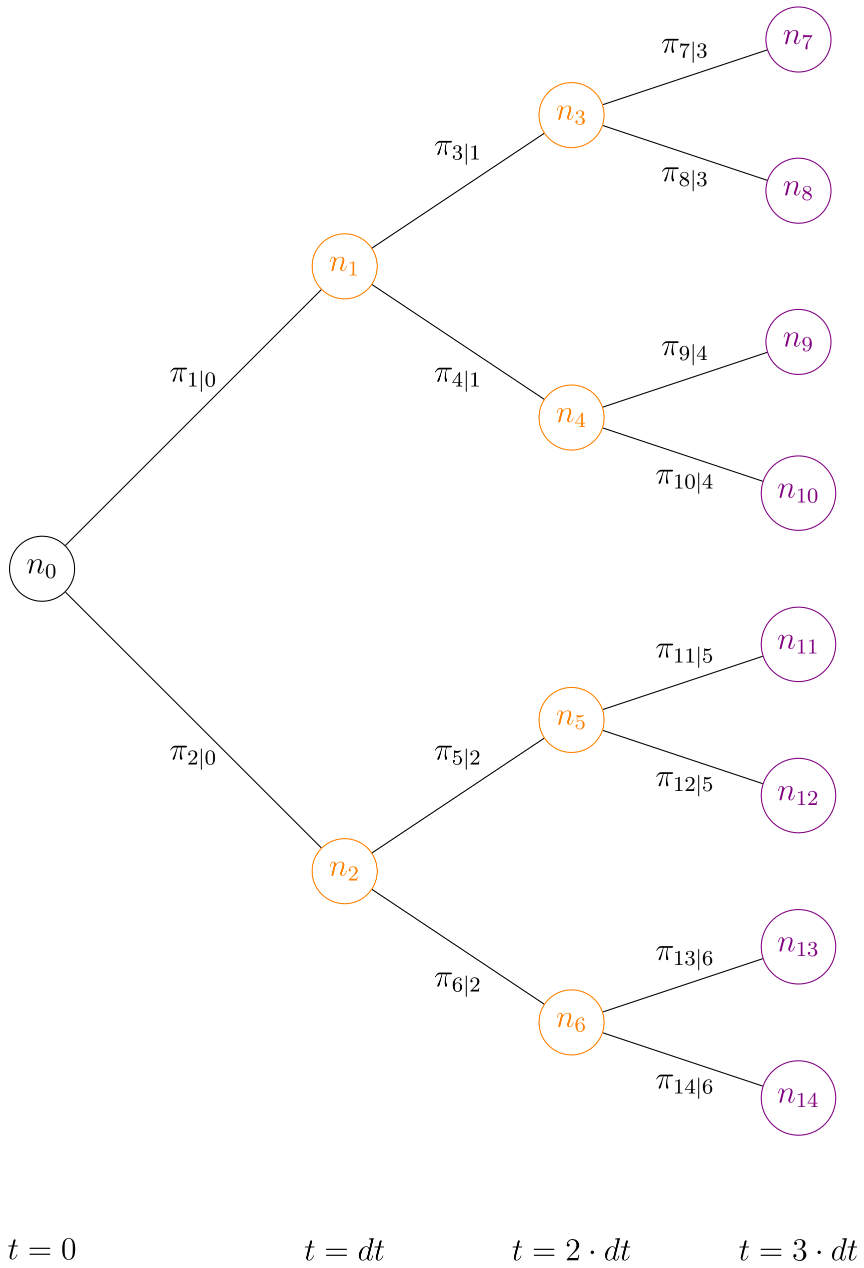


Figure 2.1: Example of a scenario tree. The orange nodes are the intermediate nodes, while the violet ones represent scenarios. The first scenario is in fact given by the path $\{n_0, n_1, n_3, n_7\}$ and has a probability equal to $\pi_{1|0} \cdot \pi_{3|1} \cdot \pi_{7|3}$. The $\pi_{p|q}$ denotes the conditional probability of node n_p given node n_q.

hedging error. In a first formulation, both upward and downward deviations are symmetrically penalized with identical weights (equal to 1). The stochastic nature of the problem is addressed using scenario trees: consequently, parameters and decision variables may depend on tree nodes $i \in \mathcal{N} = \{n_0\} \cup \mathcal{I} \cup \mathcal{S}$, where $\{n_0\}$ is the first node (which represents time $t = 0$) and $\mathcal{I}$ and $\mathcal{S}$ are, respectively, the sets of intermediate and leaf nodes; the latter, in fact, corresponds to the set of scenarios. A visual representation of a scenario tree is provided in Figure 2.1.
Hence, a standard formulation of the hedging problem is given by:

$$
\begin{aligned}
\min_{\boldsymbol{e}_+, \boldsymbol{e}_-, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}} \quad & \sum_{s \in S} \pi^s (e_+^s + e_-^s) && && (2.1)\\
\text{s.t.} \quad & x_j^{n_0} = y_j^{n_0} - z_j^{n_0} && \forall j \in \mathcal{A} && (2.2)\\
& x_j^i = x_j^{a(i)} + y_j^i - z_j^i && \forall j \in \mathcal{A}, i \in \mathcal{I} && (2.3)\\
& \sum_{j \in \mathcal{A}} z_j^{n_0} p_j^{n_0} (1 - c_j) - \sum_{j \in \mathcal{A}} y_j^{n_0} p_j^{n_0} (1 + c_j) - l^{n_0} + W_0 = 0 && && (2.4)\\
& \sum_{j \in \mathcal{A}} z_j^i p_j^i (1 - c_j) - \sum_{j \in \mathcal{A}} y_j^i p_j^i (1 + c_j) - l^i = 0 && \forall i \in \mathcal{I} && (2.5)\\
& \sum_{j \in \mathcal{A}} x_j^{a(s)} p_j^s - \Psi^s = e_+^s - e_-^s && \forall s \in \mathcal{S} && (2.6)\\
& x_j^i \in \mathbb{R} && \forall j \in \mathcal{A}, i \in \mathcal{N} \setminus \mathcal{S} && (2.7)\\
& y_j^i, z_j^i \geq 0 && \forall j \in \mathcal{A}, i \in \mathcal{N} \setminus \mathcal{S} && (2.8)\\
& e_+^s, e_-^s \geq 0 && \forall s \in \mathcal{S} && (2.9)
\end{aligned}
$$

The decision variables are:

- $x_j^i \ \forall j \in \mathcal{A}, \forall i \in \{n_0\} \cup \mathcal{I}$: quantity of asset j at node i after rebalancing;
- $y_j^i \ \forall j \in \mathcal{A}, \forall i \in \{n_0\} \cup \mathcal{I}$: quantity bought of asset j at node i;
- $z_j^i \ \forall j \in \mathcal{A}, \forall i \in \{n_o\} \cup \mathcal{I}$: quantity sold of asset j at node i.

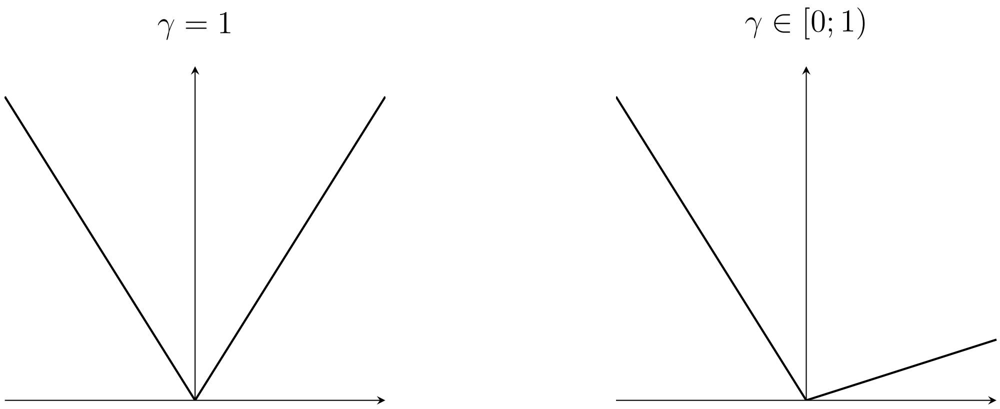
All these quantities interact in inventory balance constraints $(2.2) - (2.3)$, stating that the quantity hold for an asset after rebalancing must be equal to the hold quantity before the adjustment, plus the purchased quantity and minus the sold quantity. Cash balance constraints $(2.4) - (2.5)$ guarantee that cash inflows, derived from the potential sale of the hedging assets, entirely cover cash outflows resulting from the purchase of assets and the presence of some liabilities l^i. Transaction costs $c_j \ \ \forall j \in \mathcal{A}$ are included and handled in a proportional manner; the price of asset j in node i is denoted by p_j^i. At the initial time $t = 0$, it is assumed that an initial wealth W_0 is available to construct the portfolio and typically corresponds to the initial price of the target asset to be hedged. Alternatively, the optimization solver may be allowed to choose the necessary initial wealth. In this case, constraint (2.4) should be modified as

$$\sum_{j\in\mathcal{A}} z_j^{n_0} p_j^{n_0}(1-c_j) - \sum_{j\in\mathcal{A}} y_j^{n_0} p_j^{n_0}(1+c_j) - l^{n_0} \leq 0. \tag{2.10}$$

Note that scenarios are not included in the previous variables and constraints, since no rebalancing happens at maturity. However, there are other decision variables at maturity, i.e. $e_+^s, e_-^s \ \ \forall s \in \mathcal{S}$, which are expressly defined by constraint (2.6): they represent the positive and negative part of the difference between the derivative payoff Ψ^s in scenario s and the value of the hedging portfolio at maturity, where the quantities are those chosen at the previous node $a(s)$ and kept until maturity. An optional penalty coefficient γ may be introduced to regulate the impact of deviations in the objective function, leading to the asymmetric formulation. For instance, to emphasize loss penalization, positive errors $\boldsymbol{e}_+$ can be scaled by a factor $\gamma \in [0,1)$:

$$\sum_{s\in\mathcal{S}} \pi^s(\gamma \cdot e_+^s + e_-^s). \tag{2.11}$$

The objective still remains to replicate the derivative's payoff at maturity, but positive deviations (profits) are penalized less than negative ones (losses). Formulation (2.1) is equivalent to (2.11) when $\gamma = 1$. The penalization differences between symmetric and non-symmetric objective functions are graphically represented in Figure 2.2.



(a) Symmetric penalization on both positive and negative deviations.

(b) Asymmetric penalization: lower penalty on the positive error.

Figure 2.2: Visual representation of symmetric and asymmetric objective function.

Among all decision variables, only the initial quantities $x_j^{n_0} \ \ \forall j \in \mathcal{A}$ have a practical relevance, since they define the portfolio's composition at the current time.
Such optimization problem is solved at each rebalancing stage, considering the

evolved market conditions. Thus, constraint (2.2) has to take into account also the hedging assets holdings $\bar{x}_j$ of the previous stage's portfolio, becoming

$$x_j^{no} = \bar{x}_j + y_j^{no} - z_j^{no} \qquad \forall j \in \mathcal{A}.$$

Moreover, the cash balance constraint for the initial node (corresponding to the current time) no longer allows for an additional wealth W_0 (wich is specific for time $t = 0$, when the portfolio is built for the first time) and thus takes the form of (2.5), i.e.

$$\sum_{j\in\mathcal{A}} z_j^{no} p_j^{no}(1 - c_j) - \sum_{j\in\mathcal{A}} y_j^{no} p_j^{no}(1 + c_j) - l^{no} = 0. \tag{2.12}$$

2.2 Non-self-financing variant

Typically, hedging strategies are self-financing: the portfolio constructed at time t to be held up to time $t+1$ (which has value $V_P^{t+1}(t)$ at time t) has to be entirely financed by the current wealth $V_P^t(t)$, which represents the current value of the portfolio constructed at the previous time and kept until t. The self-financing

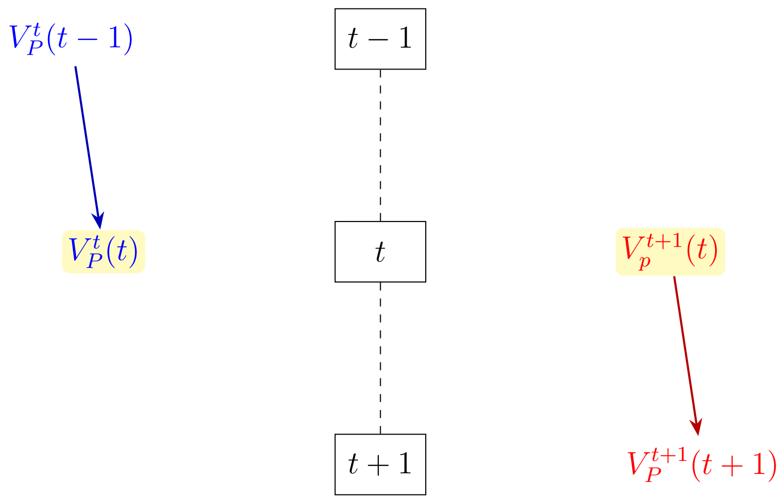


Figure 2.3: Visual representation of what self-financing portfolio means; elements of the same colour correspond to the same portfolio, but evaluated in different time instants. The two highlight quantities are those which must be equal in a self-financing framework and represent two different portfolios, but with the same value at time t.

aspect is included in the previous optimization problem in constraint (2.5) (and in (2.12) for subsequent optimization problems). However, a relaxed problem can be considered by allowing input cash flows at the root node of each scenario tree used for optimization at each rebalancing time, excluding time $t = 0$. This strategy would be not self-financing in the current timestep, but it continues to ensure the absence of inflows in the simulated future path. Formally, these considerations are summarized in the following modifications:

- (2.4), i.e. the cash balance constraint for the root node n_0, permits to receive incoming funds:

$$\sum_{j \in \mathcal{A}} z_j^{n_0} p_j^{n_0} (1 - c_j) - \sum_{j \in \mathcal{A}} y_j^{n_0} p_j^{n_0} (1 + c_j) - l^{n_0} \leq 0;$$

- for the same scenario tree, cash balance constraints for intermediate nodes $i \in \mathcal{I}$ are not modified, i.e. (2.5) is not varied.

All additional costs are referred to as *rebalancing costs*.
While this variant may seem excessively expensive, the initial formulation of the problem can, in some cases, be conservative: the only funds available for rebalancing come from the initial target asset price received by the bank at the time of sale. If this amount proves to be insufficient to fully finance the hedging strategy, the resulting performance could be suboptimal. Allowing for portfolio financing along the way may be an effective solution to improve hedging at maturity. Clearly, this comes at a cost, but it could be well worth it.
In section 6.3 the self-financing variant and the not self-financing modified version of the problem are compared in terms of performance metrics.

2.3 Withdrawal-adjusted problem

Having established that the cash balance constraint (2.4) may prove too conservative, it is possible to relax it in the opposite sense as well. Excluding the initial rebalancing step, where the portfolio is actually constructed, the constraint can be modified for the problem solved in the next rebalancing steps in order to allow for the withdrawal of surplus funds from the hedging strategy. This approach is consistent with reality: a bank may reallocate the unused funds from one strategy to support other operations.
Thus, constraint (2.4) when the root node n_0 is not the first stage becomes

$$\sum_{j \in \mathcal{A}} z_j^{n_0} p_j^{n_0} (1 - c_j) - \sum_{j \in \mathcal{A}} y_j^{n_0} p_j^{n_0} (1 + c_j) - l^{n_0} \geq 0,$$

i.e. the total amount of money generated from selling hedging assets is greater than the cash needed to purchase assets and cover liabilities, making the surplus available for withdrawal by the bank. This strategy is still considered a sort of self-financing one, since no money is added to rebalance the hedging portfolio. Lastly, both forms of relaxation can be considered together. This yields to a non-self-financing problem with the possibility to withdraw money, resulting in no constraint for node n_0 (excluding the first time step): if the strategy allows for both adding and withdrawing funds, cash flows are not subject to any constraints.

2.4 Super-replication

An alternative way to handle the replication problem is through *super-replication*. In simple terms, this approach aims to build a portfolio at the lowest possible cost while guaranteeing that its value at maturity is sufficient to cover the payoff of the derivative to be hedged. The optimization problem is then

$$\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} \quad \sum_{j\in\mathcal{A}} y_j^{n_0} p_j^{n_0}(1+c_j) - \sum_{j\in\mathcal{A}} z_j^{n_0} p_j^{n_0}(1-c_j) \tag{2.13}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{A}} x_j^{a(s)} p_j^s \geq \Psi^s - F \qquad \forall s \in \mathcal{S} \tag{2.14}$$

$$x_j^{n_0} = y_j^{n_0} - z_j^{n_0} \qquad \forall j \in \mathcal{A} \tag{2.15}$$

$$x_j^i = x_j^{a(i)} + y_j^i - z_j^i \qquad \forall j \in \mathcal{A}, i \in \mathcal{I} \tag{2.16}$$

$$\sum_{j\in\mathcal{A}} z_j^i p_j^i (1-c_j) - \sum_{j\in\mathcal{A}} y_j^i p_j^i (1+c_j) = 0 \qquad \forall i \in \mathcal{I} \tag{2.17}$$

$$x_j^i \in \mathbb{R} \qquad \forall j \in \mathcal{A}, i \in \mathcal{N} \setminus \mathcal{S} \tag{2.18}$$

$$y_j^i, z_j^i \geq 0 \qquad \forall j \in \mathcal{A}, i \in \mathcal{N} \setminus \mathcal{S} \tag{2.19}$$

The objective function aims to minimize the excess cost required to build the portfolio at time $t = 0$, which corresponds to the gap between the total outgoing and incoming cash flows at the beginning of the hedging horizon. The constraint that should characterize super-replication would be

$$\sum_{j\in\mathcal{A}} x_j^{a(s)} p_j^s \geq \Psi^s \qquad \forall s \in \mathcal{S},$$

i.e. the portfolio value at maturity, even after rebalancing during the process, is required to always exceed the payoff of the short position in the derivative to be hedged. Clearly, this is a highly conservative and rigid condition. To relax the problem, a liquidity fund F can be introduced, allowing for a discrepancy between the portfolio value and the payoff at maturity, but within a certain limit equal to F. This results in constraint (2.14).
The problem is still self-financing (constraint (2.17)) and has to satisfy inventory constraints (2.16) − (2.15).
Ensuring that the hedging strategy never leads to a loss for the bank makes this

approach very expensive to adopt, forcing the bank to sell the derivative at a higher price, which would greatly reduce the number of potential buyers. While this may not be the most practical and applicable hedging method, it can still be a useful benchmark for comparison with the strategies previously described.

2.5 Benchmarking stochastic optimization: Delta Hedging

This section explores an alternative approach to hedging, in contrast to stochastic optimization-based techniques: *delta hedging.* The key idea under delta hedging is to construct a proper portfolio that will be insensitive to changes in the underlying stock price. Considering a stock option, the quantity

$$\frac{\Delta D}{\Delta S},$$

i.e. the ratio of the change in the value of the derivative ΔD to the correspondent change in the underlying price ΔS, is known as the *delta of the stock option* and it is usually denoted by Δ. When considering a small perturbation dS in the underlying price, the resulting change in the option price is given by the derivative of its value D with respect to S, i.e.

$$\Delta = \frac{dD}{dS}.$$

A portfolio is called *delta-neutral* when its overall delta is zero.
The delta hedging objective is to maintain a portfolio whose overall value does not significantly fluctuate when changes in the underlying prices occur. Thus, the key aspect of this strategy consists in setting the portfolio's delta to zero and rebalancing it to maintain delta-neutrality. A typical composition of this portfolio includes the underlying asset, a short position into the option to be hedged and a bank account to manage cash flows. If $(\alpha, \beta, -1)$ represents the holdings vector of the portfolio, where each entry is referred to the position of, respectively, cash, stock and the derivative, the portfolio value $V_P(S)$ can be formalized as

$$V_P(S) = \alpha + \beta S - D(S) \tag{2.20}$$

where a bank account with current value equal to 1 is considered. If a small perturbation dS occurs, the correspondent change in the portfolio value is given by

$$\frac{dV_P(S)}{dS} = \beta - \frac{dD(S)}{dS},$$

i.e. the delta of the considered portfolio. By imposing delta-neutrality, it results that the position to be taken in the underlying stock is exactly the derivative's delta:

$$\Delta_P = \frac{dV_P(S)}{dS} = 0 \implies \beta = \frac{dD(S)}{dS}.$$

This aspect will be better exploited during the implementation of the delta hedging strategy.[1]
The critical point of this method is that it requires a well-defined option pricing model, in order to be able to explicitly compute the derivative with respect to the underlying price; as an example, the Black-Scholes-Merton model[2] will be considered for European options.
A comparison between the previously discussed stochastic optimization models and the current analysis leads to the conclusion that, in the case of delta hedging, the hedging assets consist of cash and the underlying asset, while the hedged position corresponds to the portfolio (2.20). The primary function of the bank account is to support all required cash flows during rebalancing.
Delta naturally evolves over time, requiring periodic portfolio adjustments to sustain a delta-neutral position to hedge the option. Nevertheless, frequent rebalancing with transaction costs can lead to considerable losses, which is a notable drawback of delta hedging.

[1]See Chapter 6.8 for details.

[2]See Chapter 4 for details.

Chapter 3

Stochastic models for stock prices

Stochastic optimization requires a representation of the underlying uncertainty in order to make decisions: one common approach to modeling uncertainties, particularly in financial applications, is the simulation of stock price paths. In financial markets, stock prices evolve over time in ways that are uncertain and influenced by a variety of factors. To address this randomness mathematically, stochastic models are employed: they account for uncertainty and allow the construction of a valid method for pricing derivatives.
This chapter shows two stochastic models that can be assumed in order to model the stock prices process: Geometric Brownian Motion and Moment Matching.
Generally speaking, if the value of a variable changes over time with uncertainty, the latter is said to follow a *stochastic process*. In particular, stock prices are assumed to follow a stochastic process that is:

- *continuous-variable*: the underlying variable can take any value within a certain range (and not only some discrete values, as in a discrete-variable process);
- *continuous-time*: the value of the considered variable can change at any time (i.e it is not restricted to a fixed set of discrete time points, like in discrete-time processes).

Stock prices are modeled using *Markov processes*, a class of stochastic processes where the future state depends only on the present state, disregarding the entire past history: the current value encodes all the relevant information for predicting the future. In terms of probabilities, this property (called *Markov property*) reflects the independence between the probability distribution of the future price at any future time and the past trajectory followed by prices up to the current state.

To be more precise, this means that the past path is irrelevant; nevertheless, the statistical properties of historical prices can still provide useful information for determining the characteristics of the stochastic process followed by the stock prices (e.g. its volatility).

3.1 Geometric Brownian Motion (GBM)

A more specific Markov process will be introduced now, commonly known as *Weiner Process* or *(standard) Brownian Motion*. Formally, the Weiner process $W(t)$ is characterized by four main properties:

1. it has independent increments, i.e. $W(t_1) - W(t_0), ..., W(t_m) - W(t_{m-1})$ are independent random variables for all $0 \leq t_0 < t_1 < ...t_{m-1} < t_m$ and $m \geq 1$;
2. increments are normally distributed with zero mean and variance equal to the time interval length, i.e.
$$W(t) - W(s) \sim \mathcal{N}(0, t - s);$$
3. $W(0) = 0$;
4. $W(t), t \geq 0$ are continuous functions of t.

Another key point is that the change ΔW over a small period of time Δt is $\Delta W = \varepsilon\sqrt{\Delta t}$, with $\varepsilon \sim \mathcal{N}(0,1)$. This obviously implies that

$$\Delta W = \varepsilon\sqrt{\Delta t} \sim \mathcal{N}(0, \Delta t).$$

When considering small changes which become closer to zero, the notation dW represents a Wiener process in the sense that it has the same properties of ΔW in the limit as $\Delta t \to 0$.
The *drift rate* is the mean change per unit time, while the variance per unit time is called *variance rate*.
It is possible to construct a *drifted* Brownian Motion: a variable B is said to follow a *generalized Wiener process* if it can be defined in terms of dW by

$$dB = \mu dt + \sigma dW, \tag{3.1}$$

where μ and σ are constants. Practically speaking, the σdW term adds random noise to the path followed by B (which would be $B = B_0 + \mu t$ if considering only the deterministic part of (3.1), i.e. $dB = \mu dt$). This concept is better depicted in Figure 3.1.

Defining $B(t) = x_0 + \mu t + \sigma W(t)$, the process $B(t)$ has a normal distribution at each time t:

$$B(t) \sim \mathcal{N}\left(x_0 + \mu t, \sigma^2 t\right).$$

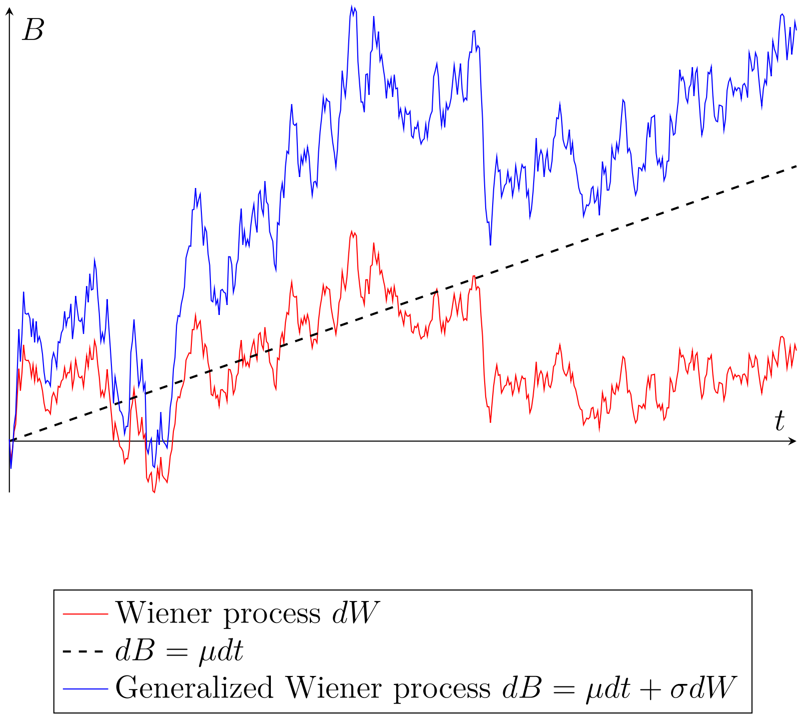


Figure 3.1: The drifted Brownian Motion is the result of adding random noise to the deterministic path described by $dB = \mu dt$.

As in the standard case, the change ΔB during a small period of time Δt is normally distributed

$$\Delta B = \mu \Delta t + \sigma \Delta W \sim \mathcal{N}\left(\mu \Delta t, \sigma^2 \Delta t\right),$$

so it has an expected drift rate (average drift per unit of time) of μ and a variance rate of σ^2.

However, a generalized Brownian Motion is not a suitable model for stock prices, since it may take negative values. Moreover, since the expected percentage return required by investors is independent from the stock's price, this model fails to effectively capture the key characteristics of stock prices. For this reason, it is necessary to introduce the *Itô process*, an extension of the generalized Brownian motion where μ and σ depend on both time t and the variable itself B. In formal notation, an Itô process can be expressed in the following way

$$dX = \mu(X, t)dt + \sigma(X, t)dW. \tag{3.2}$$

A powerful tool for understanding the behaviour of functions of stochastic variables is the *Itô's lemma*. It shows that the process of a function G of t and X, where the latter follows the process (3.2), is

$$dG = \left(\frac{\partial G}{\partial t} + \mu \frac{\partial G}{\partial X} + \frac{1}{2}\sigma^2 \frac{\partial^2 G}{\partial X^2} \right) dt + \sigma \frac{\partial G}{\partial X} dW. \tag{3.3}$$

Note that in the previous formula, μ and σ generally assume the form that defines an Itô process, i.e. they are functions (and not necessarily constant parameters). Now that all the necessary tools are at hand, it is possible to move on to address the process assumed for stock prices S, meaning that the focus is now on a process like $dS = \mu(S,t)dt + \sigma(S,t)dW$. As previously said, the expected drift rate cannot be assumed constant: a more proper assumption should state that the expected drift divided by the stock price is constant, resulting in having an expected drift rate equal to μS, i.e. $\mu(S,t) = \mu S$, for some constant parameter μ (the stock's expected rate of return). In a similar manner, the standard deviation of the change over a period of time closer to zero is assumed to be proportional to the stock price S, i.e. $\sigma(S,t) = \sigma S$, for some constant parameter σ (the volatility of the stock price). All these assumptions lead to the most commonly model used to describe stock prices, known as *Geometric Brownian Motion*:

$$dS = \mu S dt + \sigma S dW. \tag{3.4}$$

Attention is now shifted to the process followed by $Y = \ln S$, given that S follows the Geometric Brownian Motion (3.4). Exploiting Itô's lemma, dY is obtained as follows:

$$\frac{\partial Y}{\partial t} = 0 \qquad \frac{\partial Y}{\partial S} = \frac{1}{S} \qquad \frac{\partial^2 Y}{\partial S^2} = -\frac{1}{S^2}$$

$$dY = \left(\frac{\partial Y}{\partial t} + \mu S \frac{\partial Y}{\partial S} + \frac{1}{2}(\sigma S)^2 \frac{\partial^2 Y}{\partial S^2} \right) dt + \sigma S \frac{\partial Y}{\partial S} dW = \tag{3.5}$$

$$= \left[0 + \mu S \frac{1}{S} + \frac{1}{2}(\sigma S)^2 \left(-\frac{1}{S^2} \right) \right] dt + \sigma S \frac{1}{S} dW = \tag{3.6}$$

$$= \left(\mu - \frac{\sigma^2}{2} \right) dt + \sigma dW. \tag{3.7}$$

In (3.5) Ito's lemma (3.3) is applied considering the process followed by S that was previously described; partial derivatives are substituted in (3.6); in the final step, straightforward algebraic manipulations are executed to derive the final expression. The process followed by $Y = \ln S$ is therefore a generalized Wiener process with a constant drift rate $\mu - \frac{\sigma^2}{2}$ and a constant variance rate σ^2. Rewriting (3.7) in terms of $S(t)$ leads to

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

which is equivalent to the integral form

$$\int_0^t d\left(\ln S(u)\right)=\int_0^t\left(\mu-\frac{\sigma^2}{2}\right)ds+\int_0^t\sigma dW(s).$$

The solution of the previous s.d.e. is

$$\ln S(t)-\ln S(0)=\left(\mu-\frac{\sigma^2}{2}\right)t+\sigma W(t). \tag{3.8}$$

This outcome leads to two significant consequences:

- $\ln S(t)-\ln S(0)\sim\mathcal{N}\left(\left(\mu-\frac{\sigma^2}{2}\right)t,\sigma^2 t\right)$, i.e. the change in $\ln S$ between any interval time $(0,t)$ has a normal distribution with mean $\left(\mu-\frac{\sigma^2}{2}\right)t$ and variance $\sigma^2 t$. Noting that

  $$\ln S(t)-\ln S(0)=\ln\frac{S(t)}{S(0)},$$

  where the latter is the definition of log-return, the previous statement specifically means that log-return's distribution is normal.
  Equivalently

  $$\ln S(t)\sim\mathcal{N}\left(\ln S(0)+\left(\mu-\frac{\sigma^2}{2}\right)t,\sigma^2 t\right),$$

  i.e. $\ln S(t)$ is normally distributed, having mean $\ln S(0)+\left(\mu-\frac{\sigma^2}{2}\right)t$ and variance $\sigma^2 t$;

- by rearranging equation (3.8) to express $S(t)$, the price process under the historical measure μ is found:

  $$S(t)=S(0)\exp\left\{\left(\mu-\frac{\sigma^2}{2}\right)t+\sigma W(t)\right\}. \tag{3.9}$$

A deeper analysis of the first consequence leads to the conclusion that the stock price $S(t)$ at time t has a lognormal distribution[1].

[1] A variable is lognormally distributed if the natural logarithm of the variable is normally distributed.

3.2 Moment Matching

Moment matching emerges as a valid alternative for scenario generation when not assuming Geometric Brownian Motion. The purpose of this method is to maintain statistical consistency between the generated scenarios and historical (real world) data, by *matching* some statistical properties. Mean, variance, skewness and correlation (the latter only in the presence of more than one stock) are specifically chosen in the current analysis as properties to be aligned. This method relies on a least-squares minimization approach, where the objective consists in minimizing a suitable distance between the moments and correlation of the generated values and their historical counterparts.
A single-stage formulation is used, where probabilities and stock values are generated for one step, evolving from the parent node to its children nodes. Mathematically speaking, the decision variables are represented by a vector $\boldsymbol{\pi}$ of probabilities and a vector $\boldsymbol{S}$ of stock prices, each of whom has dimension equal to the number of children nodes K. Clearly, π_k denotes the probability that the stock will take the value S^k in the next step, starting from the current parent node.
As a result, this formulation involves a constrained minimization problem, subject to the constraints that the probabilities must sum up to one (3.11) and must be non-negative (3.12):

$$\min_{\boldsymbol{\pi},\boldsymbol{S}} \quad [d(\nu,\hat{\nu})]^2 + [d(\sigma,\hat{\sigma})]^2 + [d(\varsigma,\hat{\varsigma})]^2 \tag{3.10}$$

$$\text{s.t.} \quad \sum_{k\in[K]} \pi_k = 1 \tag{3.11}$$

$$\pi_k \geq 0 \qquad\qquad \forall k \in [K] \tag{3.12}$$

In (3.10), ν, σ, ς are, respectively, the mean, standard deviation and skewness of stock log-returns, estimated from historical data, and $d(\cdot,\cdot)$ represents a proper distance measure. Meanwhile,

$$\hat{\nu} = \sum_{k\in[K]} \pi_k \ln\left(\frac{S^k}{S^{a(k)}}\right) \tag{3.13}$$

$$\hat{\sigma} = \sqrt{\sum_{k\in[K]} \pi_k \left(\ln\left(\frac{S^k}{S^{a(k)}}\right) - \hat{\nu}\right)^2} \tag{3.14}$$

$$\hat{\varsigma} = \sum_{k\in[K]} \pi_k \left[\left(\ln\left(\frac{S^k}{S^{a(k)}}\right) - \hat{\nu}\right)\frac{1}{\hat{\sigma}}\right]^3, \tag{3.15}$$

where $S^{a(k)}$ is the stock value in the parent node (which is known), while S^k is the stock value in the k-th child node, i.e. the k-th component of $\boldsymbol{S}$. The single stage

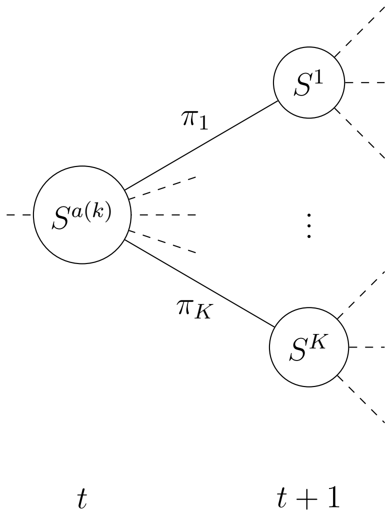


Figure 3.2: Visual representation of a single stage. Note that the value $S^{a(k)}$ in the root node is the same for each child node k. π_k is the probability of having S^k in the next time step, starting from the specific parent node $a(k)$.

is depicted in Figure 3.2.
When simulating multiple stocks, the correlation $\boldsymbol{\rho}$ among them can be incorporated as an additional property to be matched, resulting in the addition of a term $[d(\boldsymbol{\rho}, \hat{\boldsymbol{\rho}})]^2$, with a proper measure of distance, in (3.10). Clearly, with more than one stock, the properties of all stocks must be considered in the optimization problem, i.e. (3.13) − (3.14) − (3.15) must hold for all stocks $j \in [m]$ as follows:

$$\hat{\nu}_j = \sum_{k\in[K]} \pi_k \ln\left(\frac{S_j^k}{S_j^{a(k)}}\right) \qquad \forall j \in [m] \tag{3.16}$$

$$\hat{\sigma}_j = \sqrt{\sum_{k\in[K]} \pi_k \left(\ln\left(\frac{S_j^k}{S_j^{a(k)}}\right) - \hat{\nu}_j\right)^2} \qquad \forall j \in [m] \tag{3.17}$$

$$\hat{\varsigma}_j = \sum_{k\in[K]} \pi_k \left[\left(\ln\left(\frac{S_j^k}{S_j^{a(k)}}\right) - \hat{\nu}_j\right)\frac{1}{\hat{\sigma}_j}\right]^3 \qquad \forall j \in [m] \tag{3.18}$$

$$\hat{\rho}_{j,l} = \frac{1}{\hat{\sigma}_j\hat{\sigma}_l}\hat{\text{cov}}\left(\ln\left(\frac{S_j^k}{S_j^{a(k)}}\right), \ln\left(\frac{S_l^k}{S_l^{a(k)}}\right)\right) = \tag{3.19}$$

$$= \frac{1}{\hat{\sigma}_j\hat{\sigma}_l}\sum_{k\in[K]} \pi_k \left[\ln\left(\frac{S_j^k}{S_j^{a(k)}}\right) - \hat{\nu}_j\right]\left[\ln\left(\frac{S_l^k}{S_l^{a(k)}}\right) - \hat{\nu}_l\right] \qquad \forall j,l \in [m] \tag{3.20}$$

As a result $\hat{\boldsymbol{\nu}}, \hat{\boldsymbol{\sigma}}, \hat{\boldsymbol{\varsigma}}$ and their historical counterparts become vectors of dimension m, whereas $\hat{\boldsymbol{\rho}}$ and $\boldsymbol{\rho}$ are structured as matrices.
The optimization problem (3.10) is typically non-convex, thus its solution may be only a local one. Nevertheless, in this context it is sufficient to obtain a solution whose properties match or closely approximate the provided characteristics, even if better solutions may exist: an objective value close to zero guarantees that the match has been done.

3.3 Arbitrage-free scenarios

When generating scenario trees for stock prices, attention must be paid to arbitrage opportunities. Arbitrage can be mathematically defined in multiple ways, some of which will be discussed later. In simple terms, an arbitrage strategy is a risk-free method of generating sure profit without requiring any initial capital. This is equivalent to asses that a money-making machine exists, which in reality should not be acceptable. In practice, an arbitrage opportunity could arise, but it would exist only for a very short time period, as arbitrageurs would exploit it, driving prices back in line and eliminating the arbitrage opportunity itself. Therefore, such opportunities are typically assumed to be absent in real markets; thus, it is crucial to ensure that scenario trees are arbitrage-free, reflecting the real-market assumption.
Two methods for accounting for arbitrage in scenario tree generation are presented in detail in the following sections.

3.3.1 First approach: absence of dominant strategies

To better formalize, the concept of a *dominant strategy* is introduced: a trading strategy $\tilde{\boldsymbol{h}}$ with value process $\tilde{V}$ is dominant if there exists another strategy $\check{\boldsymbol{h}}$ with value process $\check{V}$ such that

$$\begin{cases} \tilde{V}(0) = \check{V}(0) \\ \tilde{V}(T,\omega) > \check{V}(T,\omega) \quad \forall \omega \in \Omega \end{cases}$$

i.e. $\tilde{\boldsymbol{h}}$ and $\check{\boldsymbol{h}}$ are two strategies that have the same value now (at $t = 0$) but the former dominates the latter state by state w in the sample space Ω, in the future at time T. Note that the values at time $t = 0$ do not depend on scenarios, since the initial state is known. Practically speaking, by purchasing the dominant strategy and selling the dominated one, the initial cash flow is zero, but this ensure a guaranteed profit in the future; this statement could be written as follows, since it is possible to take linear combination of trading strategies:

$$\begin{cases} V(0) = 0 \\ V(T,\omega) > 0 \quad \forall \omega \in \Omega \end{cases}. \tag{3.21}$$

An alternative notion of dominant strategy can be defined, by firstly assuming the existence of a bank account in the considered financial market (and this is the case, viewing the cash as a bank account). In this framework, a dominant strategy exists if and only if there exists a trading strategy which satisfies

$$\begin{cases} V(0) < 0 \\ V(T,\omega) \geq 0 \quad \forall \omega \in \Omega \end{cases}. \tag{3.22}$$

It can be shown that formulations (3.21) and (3.22) are equivalent, but only if the market includes a bank account. For optimization purposes, (3.22) is preferred as it defines the inequalities $V(T,\omega) \geq 0$ which characterize a closed set, whereas the strict inequalities in (3.21) would lead to an open set, which would not guarantee the existence of minimum or maximum.
Thus, based on formulation (3.22), the following linear programming (LP) problem can be constructed to find the strategy with the lowest possible initial value while ensuring a non-negative final value:

$$\begin{aligned} \min_{\boldsymbol{h}} \quad & \boldsymbol{v}^T \boldsymbol{h} \\ \text{s.t.} \quad & Z\boldsymbol{h} \geq 0 \end{aligned}$$

$\boldsymbol{v}$ contains the initial values for each market element (bank account included), while Z is the matrix of future values of each market element, in each scenario $\omega \in \Omega$. This problem is certainly feasible, as it admits the strategy $\boldsymbol{h} = 0$ as a solution. However, it might be unbounded below. By recalling the relationships between an LP problem and its dual, the latter's feasibility provides useful information about the primal. In particular, if the dual problem

$$\max_{\boldsymbol{y}} \quad \boldsymbol{0}^T \boldsymbol{y} \tag{3.23}$$

$$\text{s.t.} \quad Z^T \boldsymbol{y} = \boldsymbol{v} \tag{3.24}$$

$$\boldsymbol{y} \geq 0 \tag{3.25}$$

is feasible, its objective value is just zero and equal to the primal objective value. Then, a dominant strategy does not exist: according to (3.22) the initial value has to be strictly negative.
In the specific case of scenario trees, the problem $(3.23) - (3.24) - (3.25)$ is solved

at each time step. In particular, it becomes:

$$\max_{\boldsymbol{y}} \quad \sum_{k\in[K]} 0\cdot y_k \tag{3.26}$$

$$\text{s.t.} \quad \sum_{k\in[K]} y_k S_j^k = S_j \qquad \forall j \in [m] \tag{3.27}$$

$$\mathrm{e}^{r\cdot dt} \sum_{k\in[K]} y_k = 1 \tag{3.28}$$

$$y_k \geq 0 \qquad \forall k \in [K] \tag{3.29}$$

For notational convenience, the superscript $a(k)$ has been omitted in $S_j^{a(k)}$ when considering the stock value at the root node (which is the same predecessor node for each child node k). Constraint (3.24) has been split into two constraints, (3.27) for stock prices and (3.28) for the bank account, where dt represents the time interval between the parent node and its children. A closer examination of these constraints reveals that the absence of dominant strategies results in a linear and non-negative pricing functional, which expresses the current value of an asset as a linear combination of future values across possible future scenarios. It is essential to point out that this is purely a feasibility problem: if a solution exists, then dominant strategies do not arise. The solution itself has no practical relevance.
Technically, $\boldsymbol{y}$ is not a vector of probabilities. However, considering (3.28), it may be interpreted as a probability measure if it is rescaled:

$$\boldsymbol{\pi} = \mathrm{e}^{r\cdot dt}\boldsymbol{y},$$

so that

$$\sum_{k\in[K]} \pi_k = 1$$

$$\pi_k \geq 0 \qquad \forall k \in [K].$$

This rescaling applied to (3.27) yields a relationship involving discounted processes, i.e.

$$\sum_{k\in[K]} \pi_k \frac{S_j^k}{e^{r\cdot dt}} = S_j \qquad \forall j \in [m].$$

Denoting by $\mathbb{E}_{\mathbb{Q}}$ the expectation under the probability measure defined by $\boldsymbol{\pi}$, the previous equation is equivalent to

$$S_j = \mathbb{E}_{\mathbb{Q}}\left[\frac{\boldsymbol{S}_j^*}{e^{r\cdot dt}}\right] \qquad \forall j \in [m],$$

which means that the expected value under $\mathbb{Q}$ of the discounted future value process $\frac{\boldsymbol{S}_j^*}{e^{r\cdot dt}}$ of stock j is constant (and equal to its current value S_j). This is known as *martingale property*. $\boldsymbol{S}_j^*$ contains the children nodes' prices for each stock j: $\boldsymbol{S}_j^* = (S_j^1, ..., S_j^K)$.

3.3.2 Second approach: no arbitrage constraints

Arbitrage strategies are, in reality, a less restrictive concept than dominant strategies. Starting from (3.21) and weakening it by requiring that the future values have to be strictly positive in at least one future scenario (and not in *all* of them) and non-negative in the remaining states, the *arbitrage opportunity* definition is set up:

$$\begin{cases} V(0) = 0 \\ V(T,\omega) \geq 0 \qquad \forall \omega \in \Omega \; . \\ \mathbb{E}\left[V(T,\omega)\right] > 0 \end{cases} \tag{3.30}$$

This formulation can be exploited to construct an optimization problem: if the latter is infeasible, then arbitrage opportunity cannot arise. This occurs when the corresponding dual problem is unbounded above; the dual variables, when rescaled to sum up to one, define a strictly positive probability measure $\mathbb{Q}$, known as the *risk-neutral measure*[2]. In a similar way as in the previous section, it is obtained that, under this martingale measure, the current value of a trading strategy is just the discounted expected value of its future value. The key point is that there are no arbitrage strategies if and only if a strictly positive martingale probability measure can be found.

All these concepts can be leveraged to formulate constraints that ensure the absence of arbitrage in scenario trees. Once prices at the children nodes are generated from the parent node through Geometric Brownian Motion, they can be checked and adjusted if needed to eliminate arbitrage. A least squares minimization problem can be employed to find the closest arbitrage-free values $\boldsymbol{S}_j^*$ to those initially generated by GBM $\tilde{\boldsymbol{S}}_j^*$, for each stock j:

$$\min_{\boldsymbol{S}_j^*} \quad \sum_{k\in[K]} \left(S_j^k - \tilde{S}_j^k\right)^2 \tag{3.31}$$

$$\text{s.t.} \quad S_j^{\vartheta} e^{-r\cdot dt} \geq S_j + \alpha \tag{3.32}$$

$$S_j^{\theta} e^{-r\cdot dt} \leq S_j - \alpha \tag{3.33}$$

having:

- $\vartheta = \arg\max_{k\in K} \tilde{S}_j^k$;
- $\theta = \arg\min_{k\in K} \tilde{S}_j^k$;
- α is a small strictly positive number.

[2]Under this measure, the expected return of risky assets is exactly equal to the risk-free rate r.

According to constraints (3.32) − (3.33), the highest discounted price among the newly generated values must exceed the parent node price, while the lowest must fall below it. This ensures that there is at least one value above and one below the mean (i.e. S_j). The previous optimization problem is shown in [1].
A different way to guarantee that there is at least one value above and one below the mean, instead of using constraints (3.32) − (3.33), is to impose that the return of S_j^{ϑ} has to be greater than a small α and that of S_j^{θ} has to be less than $-\alpha$, i.e.

$$\frac{S_j^{\vartheta} - S_j}{S_j} \geq \alpha$$

$$\frac{S_j^{\theta} - S_j}{S_j} \leq -\alpha$$

Chapter 4

Options and exotic derivatives

Derivatives are financial instruments whose value is explicitly defined in a contract as a function of another variable. If the derivative is written on a stock share with price $S(t)$ at time t, the latter is called its *underlying asset*. Derivatives may rely on multiple underlying variables and they differ in terms of the function defining the payoff.
In this framework, vanilla options are treated as hedging instruments (although they will also be subjected to hedging, providing a benchmark to assess the effectiveness of the adopted strategies), while Asian options, barrier options and worst-performance derivatives are regarded as exotic instruments requiring hedging.

4.1 Pricing derivatives: risk-neutral valuation

In order to properly address derivatives, it is firstly necessary to present the pricing principle concerning their valuation, known as *risk-neutral valuation*. This pricing tool relies on the assumption that all investors behave like they are risk-neutral, meaning that they do not require an increasing compensation for bearing an increased risk: the expected return on all investments is just the risk-free rate r. In terms of pricing, the key implication is that the present value of any future cash flow is obtained by discounting its expected value at the risk-free interest rate. Going into more details, the procedure of valuing a generic derivative which provides a payoff $\Psi(T)$ at time T is structured as follows:

1. assume that the expected return from the underlying asset(s) is the risk-free rate r (practically, this means that $\mu \equiv r$);

2. compute the derivative expected payoff at T, i.e.

$$\mathbb{E}_{\mathbb{Q}}\left[\Psi(T)\right],$$

considering a risk-neutral probability measure $\mathbb{Q}$;

3. discount the expected payoff at the risk-free rate r from T to the current time t, obtainig the current value of the considered derivative

$$\Pi(t,T) = \frac{\mathbb{E}_{\mathbb{Q}}\left[\Psi(T)\right]}{\mathrm{e}^{r\cdot(T-t)}}. \tag{4.1}$$

Note that a continuously compounded rate r has been considered.

A *risk-neutral world* represents the framework where this principle holds. Clearly, the real world is not risk-neutral, but the assumption underlying this principle allows for the correct pricing of derivatives in any world, not just in a risk-neutral setting.

4.2 Vanilla options

Two types of options are considered:

- *call* options, which give the holder the right, but not the obligation, to buy the underlying asset from the option writer, in the future and at a fixed strike price K;
- *put* options, where the option holder has the right, but not the obligation, to sell the underlying asset to the option writer, in the future and at a fixed strike price K.

Unlike the writer, the holder has the right to make a choice. This distinction introduces an initial cost for options, reflecting the writer's compensation for the obligation to comply with the holder's decision. Thus, options are asymmetric derivatives and their acquisition entails a cost, unlike linear contracts. As suggested by the previous explanation, every option contract involves two parties. On one side, the option writer, who has sold the option, is said to hold the *short position*; he/she receives cash up front, but could have potential liabilities later. On the other side, the option holder, i.e. the one who has bought the option, holds the *long position*. For better understanding, the long position is generally linked to the contract side that profits from an increase in the instrument's value, whereas the short position benefits when the instrument loses value.
Options can be distinguished by their exercise date. American options can be exercised at any time before maturity (early exercise) or at maturity itself. On the

contrary, European options allow exercise only at maturity. In this context, the focus is exclusively on European-style options. Having the underlying stock value $S_T = S(T)$ at maturity T, from the holder viewpoint the payoff of an European call option is

$$\max\{0, S_T - K\}, \tag{4.2}$$

which reflects the fact that the option will be exercised if $S_T > K$ and will not be exercised if $S_T \leq K$. Similarly, the holder of an European put option will have a payoff given by

$$\max\{0, K - S_T\}, \tag{4.3}$$

since he will exercise the put option if $S_T < K$. The payoffs for short positions are the negative counterparts of those for long positions, as the two positions are fundamentally opposite. As a result, the payoff of a short position into a call is

$$-\max\{0, S_T - K\} = \min\{0, K - S_T\},$$

while a short position into a put has a resulting payoff given by

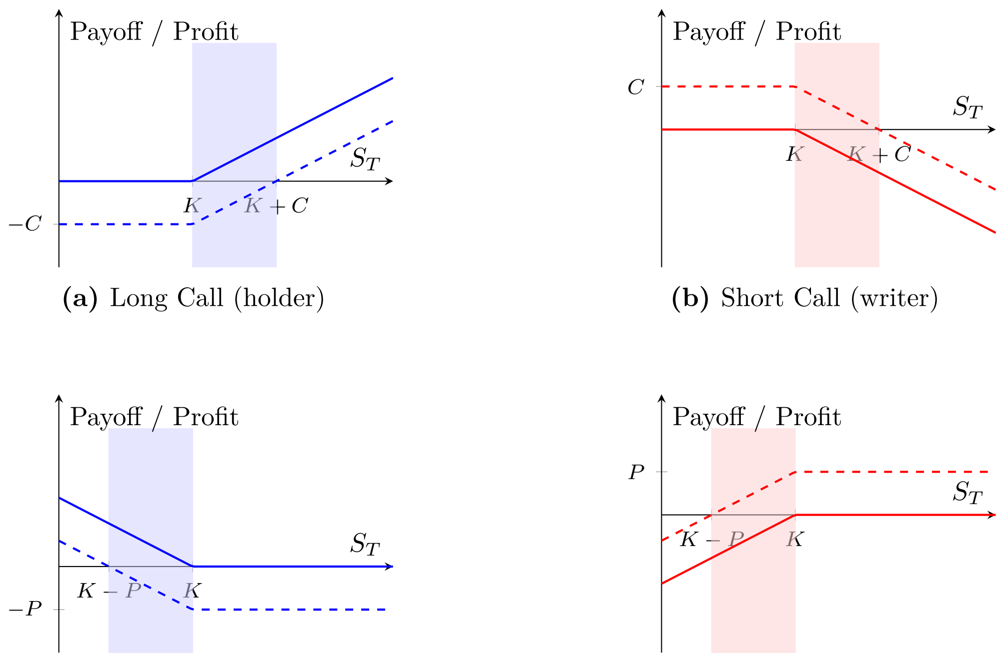
$$-\max\{0, K - S_T\} = \min\{0, S_T - K\}.$$

All these characteristics are graphically shown in Figure 4.1 by the solid lines.
Due to the presence of an initial price, payoff and profit do not coincide as they do in linear contracts. Instead, they differ by an amount equal to the initial price. In a similar manner, as discussed earlier, the short position's profit or loss is the reverse of that for the long position. Table 4.1 and Figure 4.1 provide a clearer illustration of what was just explained. When the option expires worthless, the holder loses the full option premium, which is the worst possible outcome; on the other hand, the potential loss for a call writer is unbounded. Generally speaking, exercising an option does not ensure a positive profit for the holder and a loss for the short position. The final outcome depends on the underlying's value at maturity, as follows:

- a call is exercised if $S_T > K$, but if $K < S_T < K + C$ the holder will incur a loss (Figure 4.1a) and the writer will earn a profit (Figure 4.1b);

- a put is exercised if $S_T < K$, but if $K - P < S_T < K$ the holder will incur a loss (Figure 4.1c) and the writer will earn a profit (Figure 4.1d).

In terms of hedging, it is important to consider that the purchaser of a call option relies on a potential rise in the stock price, while the holder of a put option is expecting a decrease in its underlying value.



(a) Long Call (holder) **(b)** Short Call (writer)

(c) Long Put (holder) **(d)** Short Put (writer)

Figure 4.1: Payoff (solid lines) and profit (dashed lines) for long and short positions in call and put options. K is the strike price, S_T is the underlying price at maturity T, C and P are respectively the call and the put initial price. The blue shaded regions highlight the range where the holder is exercising the option, but still having a loss. The red shaded regions represent the range where the writer incurs a profit, even if the counterpart is exercising the option.

Option type	**Position**	**Payoff**	**Profit**
Call	Long	$\max\{0, S_T - K\}$	$\max\{0, S_T - K\} - C$
	Short	$-\max\{0, S_T - K\}$	$C - \max\{0, S_T - K\}$
Put	Long	$\max\{0, K - S_T\}$	$\max\{0, K - S_T\} - P$
	Short	$-\max\{0, K - S_T\}$	$P - \max\{0, K - S_T\}$

Table 4.1: Payoff and profit for European options.

4.2.1 Pricing European vanilla options: Black-Scholes-Merton model

The Black-Scholes-Merton model is a widely used approach for pricing derivatives. It is based on some assumptions:

1. the underlying stock price follows a lognormal distribution and (3.9), as described in the previous chapter, considering constant μ and σ;
2. the considered derivative is a tradeable asset;
3. short shelling is permitted;
4. there are no frictions, i.e. transaction costs or taxes;
5. the market is arbitrage free;
6. the risk-free rate r is constant and the same for all maturities.

Note that (3.9) is generally valid in real worlds, whereas in a risk-neutral world μ must be replaced by the risk-free rate r. To be more precise, from a pricing point of view the Geometric Brownian Motion followed by stock prices becomes $dS = rSdt + \sigma SdW$.
This model is characterized by

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} = rf,$$

known as *Black-Scholes-Merton differential equation*, where $f = f(S,t)$ represents the price of the derivative having S as underlying. By solving this differential equation, all the different derivatives that can be defined with S as the underlying stock will be found. The specific derivative obtained as a solution depends on the boundary condition that is used; generally, a final condition $f(S(T),T) = \Psi(S(T))$ is given, where $\Psi(S(T))$ represents the derivative payoff at maturity T.
Furthermore, according to the Feynman-Kac theorem, the solution can be expressed as

$$f(s,t) = \mathrm{e}^{-r\cdot(T-t)}\mathbb{E}^{\mathbb{Q}}\left[\Psi(S(T))|S(t)=s\right],$$

where the expectation is made under a measure such that μ becomes r, i.e. a risk-neutral measure $\mathbb{Q}$.
It is important to notice that Black-Scholes-Merton model gives an equation that must be satisfied by the price of any derivative depending on a non-dividend-paying stock. In this context, the specific case of European vanilla options is taken into account, so e.g. the boundary condition for a European call option will be

$$f(S(T),T) = \max\{S(T)-K,0\}.$$

Consequently, the Black-Scholes-Merton pricing formula which gives the price of a European vanilla call option at a generic time $t < T$ is the following:

$$C(t) = S(t)N(d_1) - K\mathrm{e}^{-r\cdot(T-t)}N(d_2)$$

with

$$d_1 = \frac{\ln\left(\frac{S(t)}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}} \tag{4.4}$$

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

$N(\cdot)$ is the c.d.f of a standard normal distribution $\mathcal{N} \sim (0,1)$.
Clearly, the pricing formula of a vanilla European-style put option

$$P(t) = K\mathrm{e}^{-r\cdot(T-t)}N(-d_2) - S(t)N(-d_1)$$

is found by adopting the same procedure.
Generally speaking, the price of a call option decreases as the strike K increases, while the price of a put option increases as the strike K increases.

4.3 Asian options

By extending the concept of vanilla options, more complex derivatives can be created, like the exotic ones, featuring different payoff structures. As a specific category of exotic options, Asian options provide a modified payoff mechanism by using the average price of the underlying asset during the life of the option, rather than only its maturity value. This results in substituting S_T with the average price S_{avg} of the underlying asset from initial time to maturity in the previously introduced payoff formulas. As an example, the payoff of an European Asian[1] call option, starting from (4.2), becomes

$$\Psi_{AC}(T) = \max\{0, S_{avg} - K\} \qquad \text{with } \; S_{avg} = \frac{1}{m}\sum_{d\in[D]} S(t_d), \tag{4.5}$$

considering $t_d,\ d \in [D]$ as dates of evaluation and $t_D \equiv T$.
Asian options offer an additional level of complexity by allowing for multiple underlying assets, an assumption that is usually invalid for vanilla options. As a

[1] An Asian option could be European or American, since *Asian* refers only to the payoff form, whereas *European* or *American* define the exercise mechanism of the option.

consequence, the payoff of an European-style Asian put option with m underlings is defined by

$$\Psi_{AP}(T) = \max\{0, K - S_{avg}\}$$

$$\text{with} \quad S_{avg} = \frac{1}{m}\sum_{j\in[m]} S_{avg,j} = \frac{1}{m}\sum_{j\in[m]}\left(\frac{1}{D}\sum_{d\in[D]} S_j(t_d)\right),$$

reflecting the dependence from both the time-average of its underlying assets' values and the average across all its underlying assets.
In the hedging framework, priority will be given to the study of options with more than one underlying, as they are regarded as more difficult and challenging to address.
From a pricing perspective, Asian options are priced in a risk-neutral setting by discounting the expected payoff under a risk-neutral measure $\mathbb{Q}$. In a nutshell, this results in computing the price at time t of a derivative with maturity T by using (4.1).

4.4 Barrier options

More complex and sophisticated derivatives will be now introduced, known as barrier options. Their payoff depends on whether the underlying asset's prices reaches a certain level, the *barrier* H, during the derivative lifetime. From a financial standpoint, these options are appealing because they are cheaper than their vanilla counterpart.
A first classification divides barrier options into two categories:

- *knock-out* options, which cease to exist when the underlying asset price hits the barrier H;
- *knock-in* options, which on contrary come into existence when the underlying asset price reaches the barrier H.

Barrier options are additionally categorized according to the relationship between the barrier level H and the initial underlying price $S(0)$:

- if $H < S(0)$, i.e. the barrier lever is below the initial stock price, the option is said to be of *down-* type;
- if $H > S(0)$, i.e. the barrier level is above the initial stock price, the option is an *up-* barrier option.

Eight barrier options are taken into account as instruments to be hedged, four calls and four puts, whose pricing formulas are presented below.

Down-and-out **and** ***down-and-in*** **call.**

Keeping in mind that the value $C(t)$ of a vanilla call option at time t is equal to the sum between the values of the corresponding down-and-out $C_{do}(t)$ and down-and-in $C_{di}(t)$ options at time t, i.e.

$$C(t) = C_{di}(t) + C_{do}(t),$$

it can be proven that:

- for $H \le K$

$$C_{di}(t) = S(t)\left(\frac{H}{S(t)}\right)^{2\lambda} N(y) - K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2} N(y-\sigma\sqrt{T-t})$$

  and

$$C_{do}(t) = C(t) - C_{di}(t);$$

- for $H \ge K$

$$C_{do}(t) = S(t)N(x_1) - K\mathrm{e}^{-r\cdot(T-t)}N(x_1-\sigma\sqrt{T-t}) - S(t)\left(\frac{H}{S(t)}\right)^{2\lambda} N(y_1) +$$
$$+ K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2} N(y_1-\sigma\sqrt{T-t})$$

  and

$$C_{di}(t) = C(t) - C_{do}(t).$$

The used parameters are

$$\begin{cases} \lambda = \dfrac{1}{\sigma^2}\left(r + \dfrac{\sigma^2}{2}\right) \\ y = \dfrac{1}{\sigma\sqrt{T-t}}\ln\left(\dfrac{H^2}{S(t)K}\right) + \lambda\sigma\sqrt{T-t} \\ x_1 = \dfrac{1}{\sigma\sqrt{T-t}}\ln\left(\dfrac{S(t)}{H}\right) + \lambda\sigma\sqrt{T-t} \\ y_1 = \dfrac{1}{\sigma\sqrt{T-t}}\ln\left(\dfrac{H}{S(t)}\right) + \lambda\sigma\sqrt{T-t} \end{cases}.$$

Up-and-out **and** ***up-and-in*** **call.**

By a similar manner, the main relation is

$$C(t) = C_{ui}(t) + C_{uo}(t),$$

since an up-and-out call is essentially a regular call option that ceases to exist when and only if the barrier is reached, an event that simultaneously triggers the beginning of the existence of the corresponding up-and-in call. Then,

- for $H \leq K$
$$C_{uo}(t) = 0 \text{ and so } C_{ui}(t) = C(t);$$

- for $H > K$
$$\begin{aligned}C_{ui}(t) = & \ S(t)N(x_1) - K\mathrm{e}^{-r\cdot(T-t)}N\left(x_1 - \sigma\sqrt{T-t}\right) + \\ & - S(t)\left(\frac{H}{S(t)}\right)^{2\lambda}[N(-y) - N(-y_1)] + \\ & + K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2}\left[N\left(-y + \sigma\sqrt{T-t}\right) - N\left(-y_1 + \sigma\sqrt{T-t}\right)\right]\end{aligned}$$

  and
$$C_{uo}(t) = C(t) - C_{ui}(t).$$

Using the same logic, the following results hold for put options.

***Down-and-out* and *down-and-in* put.**

$$P(t) = P_{di}(t) + P_{do}(t)$$

- for $H \leq K$
$$\begin{aligned}P_{di}(t) = & \ -S(t)N(-x_1) + K\mathrm{e}^{-r\cdot(T-t)}N\left(-x_1 + \sigma\sqrt{T-t}\right) + \\ & + S(t)\left(\frac{H}{S(t)}\right)^{2\lambda}[N(y) - N(y_1)] + \\ & - K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2}\left[N\left(y - \sigma\sqrt{T-t}\right) - N\left(y_1 - \sigma\sqrt{T-t}\right)\right]\end{aligned}$$

  and
$$P_{do}(t) = P(t) - P_{di}(t);$$

- for $H > K$
$$P_{do}(t) = 0 \text{ and so } P_{di}(t) = P(t).$$

Up-and-out **and** ***up-and-in*** **put.**

$$P(t) = P_{ui}(t) + P_{uo}(t)$$

- for $H \geq K$

$$P_{ui}(t) = -S(t)\left(\frac{H}{S(t)}\right)^{2\lambda} N(-y) + K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2} N(-y+\sigma\sqrt{T-t})$$

  and

$$P_{uo}(t) = P(t) - P_{ui}(t);$$

- for $H \leq K$

$$\begin{aligned} P_{uo}(t) = & -S(t)N(-x_1) + K\mathrm{e}^{-r\cdot(T-t)}N(-x_1+\sigma\sqrt{T-t}) + \\ & + S(t)\left(\frac{H}{S(t)}\right)^{2\lambda} N(-y_1) + \\ & - K\mathrm{e}^{-r\cdot(T-t)}\left(\frac{H}{S(t)}\right)^{2\lambda-2} N(-y_1+\sigma\sqrt{T-t}) \end{aligned}$$

  and

$$P_{ui}(t) = P(t) - P_{uo}(t).$$

The previously listed analytic formulas assume that S is continuously observed in order to determine if the barrier has been hit. In a discrete framework, the barrier level H should be replaced by $H\mathrm{e}^{0.582\sigma\sqrt{T/\beta}}$ for up-options and by $H\mathrm{e}^{-0.582\sigma\sqrt{T/\beta}}$ for down-options, where β is the number of times when the asset price is observed. From a payoff perspective, when a barrier option exists at maturity, its payoff is given by the correspondent formula which depends on the type of the option: e.g., a barrier call option which is active at maturity will have a payoff given by (4.2). On the other hand, if it does not exist, its payoff is trivially equal to zero.

4.5 Worst-Performance Derivatives

Some multi-asset exotic derivatives issued by Intesa Sanpaolo bank are taken into account, representing the most challenging derivatives to hedge considered in this analysis: *Standard Long Barrier Plus Worst of Certificates* (4.5.1), *Standard Long Barrier Digital Worst of Certificates* (4.5.2) and *Standard Long Autocallable Barrier Digital Worst of Certificates with memory effect* (4.5.3). As the name suggests, they are based on the *worst performance* of their underlyings during the derivative lifetime. More formally, consider m underlyings, whose values at a generic time instant t_d are $S_j(t_d)$ for all $j \in [m]$. The performance at date t_d of the asset j with an initial value $S_j(t_0)$ can be expressed as

$$P_j(t_d) = \frac{S_j(t_d) - S_j(t_0)}{S_j(t_0).}$$

Then, the worst-performing asset (i.e. the one with the lowest performance) at t_d is found by

$$w_{t_d} = \arg\min_{j \in [m]} P_j(t_d). \tag{4.6}$$

The objective of such products is to provide additional return in exchange for the risk of loss of capital. WP derivatives share some common features: an Issue Price is setted at the beginning of the contract, as well as barrier levels related to how periodically coupons are paid (if paid) and to the assessment of the maturity payoff. In particular, the cash flow at maturity depends on worst performance and is given by

$$\Psi_{\text{WP}}(T) = \begin{cases} I & \text{if } S_{w_{\text{FVD}}}(\text{FVD}) \geq b \cdot S_{w_{\text{FVD}}}(t_0) \\ I \cdot \dfrac{S_{w_{\text{FVD}}}(\text{FVD})}{S_{w_{FVD}}(t_0)} & \text{otherwise} \end{cases} \tag{4.7}$$

where:

- the underlying stocks S_j are those listed in Table 4.2;
- T represents the Expiry Date (maturity);
- FVD corresponds to the Final Valuation Date;
- w_{FVD} follows the definition (4.6), i.e. it is the worst-performing asset at the Final Valuation Date;
- b is a percentage which defines a Barrier Level when multiplied by the worst performer value $S_{w_{FVD}}(t_0)$ at the Issue Date t_0;
- I is the Issue Price.

From a pricing point of view, the WP derivatives initial price can be determined, in a risk-neutral world, as the expected discounted payoff under a risk-neutral probability measure $\mathbb{Q}$ as previously explained in section 4.1. In the specific context of pricing, the value of this contract at maturity is given by the sum between the effective maturity cash flow (i.e. $\Psi_{\text{WP}}(T)$) and all the actualized intermediate cash flows (i.e. coupons):

$$\Pi_{\text{WP}}(t_0, T) = \frac{\mathbb{E}_{\mathbb{Q}}\left[\Psi_{\text{WP}}(T) + \sum_{d \in [D]} C_{t_d} \mathrm{e}^{r \cdot (T - t_d)}\right]}{\mathrm{e}^{r \cdot (T - t_0)}}. \tag{4.8}$$

Derivative	Underlyings
WP1	EURO STOXX 50®
	FTSE® MIB® IDX
WP2	EURO STOXX® SELECT DIVIDEND 30
	FTSE ® MIB® IDX
WP3	RWE AG
	ÉLECTRICITÉ DE FRANCE SA
	IBERDROLA SA

Table 4.2: Underlyings of each WP derivative.

However, they also have peculiarities, as detailed in what follows. The derivatives are described as they were originally developed by Intesa Sanpaolo. However, for the purposes of this work, only their structural framework will be retained for the analysis phase, without also including their real data.

4.5.1 WP1: Standard Long Barrier Plus Worst of Certificates

The first considered WP derivative provides periodic payments which are unconditionally paid since they are not linked to any underlying performance: coupons are deterministic. The Periodic Amount C_{t_d} referred to the Periodic Amount Payments Dates t_d is fixed and equal to $C_{t_d} = €27.50 = C \quad \forall d \in [D]$.
On the contrary, the payoff at maturity (T = 28 March 2023) depends on worst performance, following (4.7) and considering the specific characteristics in Table 4.3.

Issue Price I	**Periodic Amount** C	**Barrier percentage** b
€1000	€27.50	50%

Table 4.3: WP1 key values.

The price at the Issue Date t_0 can be found according to (4.5). Consequently, all the bank cash flows can be summarized as follows in Figure 4.2.

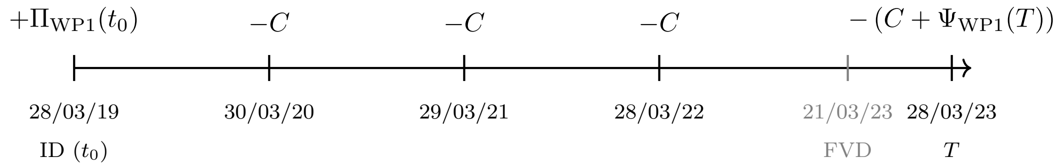


Figure 4.2: WP1 cash flows from the bank's point of view.

4.5.2 WP2: Standard Long Barrier Digital Worst of Certificates

Moving forward, the second WP derivative also involves worst performance valuation to determine whether coupons (Digital Amounts) will be paid. It is necessary the introduction of another percentage g which gives the Digital Level when multiplied by the worst performer value at t_0, i.e. the cash flow C_{t_d} at time t_d is

$$C_{t_d} = \begin{cases} G & \text{if } S_{w_{t_d}}(t_d) \geq g \cdot S_{w_{t_d}}(t_0) \\ 0 & \text{otherwise} \end{cases} \qquad \forall d \in [D]. \tag{4.9}$$

In other words, after finding the underlying with the worst performance at t_j, if its current value is greater than or equal to the Digital Level, the coupon G is paid. The formula (4.7) also defines the cash flow at maturity for this specific derivative, without any modifications. The most relevant characteristics of this product are summarized in Table 4.4. Coupon valuation dates occur quarterly starting from and including 23 June 2020 to 23 March 2026, which is also the Final Valuation Date. The contract expires on 30 March 2026.

Issue Price I	**Digital Amount** G	**Barrier percentage** b	**Digital percentage** g
€1000	€8	50%	75%

Table 4.4: WP2 key values.

4.5.3 WP3: Standard Long Autocallable Barrier Digital Worst of Certificates with memory effect

The last WP derivative taken into account has a more complex structure. Both coupon and maturity cash flows depend on the worst performance of its underlyings, as in WP2. However, it exhibits two additional and compelling properties that influence its management:

1. *memory effect*: as previously described, at each coupon date t_d the worst-performer among underlyings determines if the coupon is paid or not at the current time by comparing its current value with the Digital Level. Additionally, if the coupon is paid, the bank has to settle all the previous unpaid coupons; otherwise, if the current value of the underlying with the worst performance is below the Digital Level, no payment will occur at t_d;

2. *early redemption (autocallability)*: this product has an additional set of relevant dates, called Early Redemption Valuation Dates, when early redemption could happen if the current value of the worst performer is higher than or equal to its initial value. In this case, the derivative will be redeemed and the bank will have to pay the Early Redemption Amount ERA to the investor. As a consequence of this event, no other cash flow will be occur: the contract is expired.

Taking all these aspects into account, cash flows formula becomes

$$C_{t_d} = \begin{cases} G + \sum_{k \in U_d} G + ERA \cdot \mathbb{1}_{E_R(t_d)} & \text{if } S_{w_{t_d}}(t_d) \geq g \cdot S_{w_{t_d}}(t_0) \text{ and no early} \\ & \text{redemption has occurred in } (t_0; t_{d-1}] \\ 0 & \text{otherwise} \end{cases}, \tag{4.10}$$

where $U_d = \{k \in [z+1; d-1] : C_{t_k} = 0, C_{t_z} \neq 0\}$ refers to the dates of previous coupons that have not been paid since the last non-zero cash flow. Clearly, if t_d is the first cash flow, it is necessary to collect all coupons backwards in time until t_0. Meanwhile $E_R(t_d) = \{\text{early redemption occurs exactly at } t_d\}$ is related the to early redemption event: the multiplication by its indicator function is useful according to the fact that if early redemption takes place at t_d, the Early Redemption Amount ERA will also be paid. If the value of the worst performer does not exceed the Digital Barrier, or if an early redemption event has already occurred before t_d, the cash flow C_{t_d} will be zero: in the latter case, the contract has expired; thus, there will be no future payments.
Early redemption also affects the cash flow at maturity:

$$\Psi_{\text{WP3}}(T) = \begin{cases} 0 & \text{if early redemption have occurred until } T \\ \Psi_{\text{WP}}(T) & \text{otherwise} \end{cases}. \tag{4.11}$$

WP3's Digital Valuation Dates are scheduled every six months, starting on 10 May 2022 and ending on 14 November 2024, which is also the Final Valuation Date.

Moreover, all these dates, excluding the final one, coincide with the Early Redemption Valuation Dates.
This product's specific features are shown in Table 4.5. Its Issue Date is 16 November 2021.

Issue Price I	**Periodic Amount** G	**Early Redempition Amount** ERA	**Barrier percentage** b	**Digital percentage** g
€1000	€36.50	€1000	60%	60%

Table 4.5: WP3 key values.

Practically speaking, the valuation dates of all WP derivatives slightly differ from the ones when the payments actually take place. For the purposes of this thesis, however, it can be assumed that the valuation dates coincide with the actual cash flow dates. As a result, FVD and T are also considered to be the same.

Chapter 5

Code Structure

A well-structured Python code framework supports all the results obtained in this thesis. An object-oriented programming approach was implemented for the construction of market elements. The allowed assets are those described in previous sections, i.e.

- stocks;
- options: vanilla, Asian, barrier and worst-performance type; calls and puts are considered, for a total of two European-style, two Asian-style, eight barrier type options, in addition to the three worst-performance derivatives;
- cash.

An additional class called `MultiStock` is used to store information about the correlation among the various stocks present in the market. Since each option can have one or more underlying assets, the relationship between the option and stocks classes is essential. In particular, European and barrier options are written on a single underlying, while Asian options and worst-performance derivatives involve multiple underlyings, thus they are connected to the `MultiStock` class rather than the `Stock` one.
The `Cash` class contains the risk-free rate, which is used by each option type.
`WP1` is the most abstract class among the three WP derivatives. `WP2` inherits its features and adds the stochastic nature of the coupons. `WP3` extends `WP2` by adding the possibility of early redemption. A summary diagram of the asset classes and their relationships is presented in Figure 5.1.
After selecting the market components, their statistical properties (mean, variance, correlation, and skewness of log-returns) are estimated based on historical data of the period from 05/01/2022 to 05/01/2024.

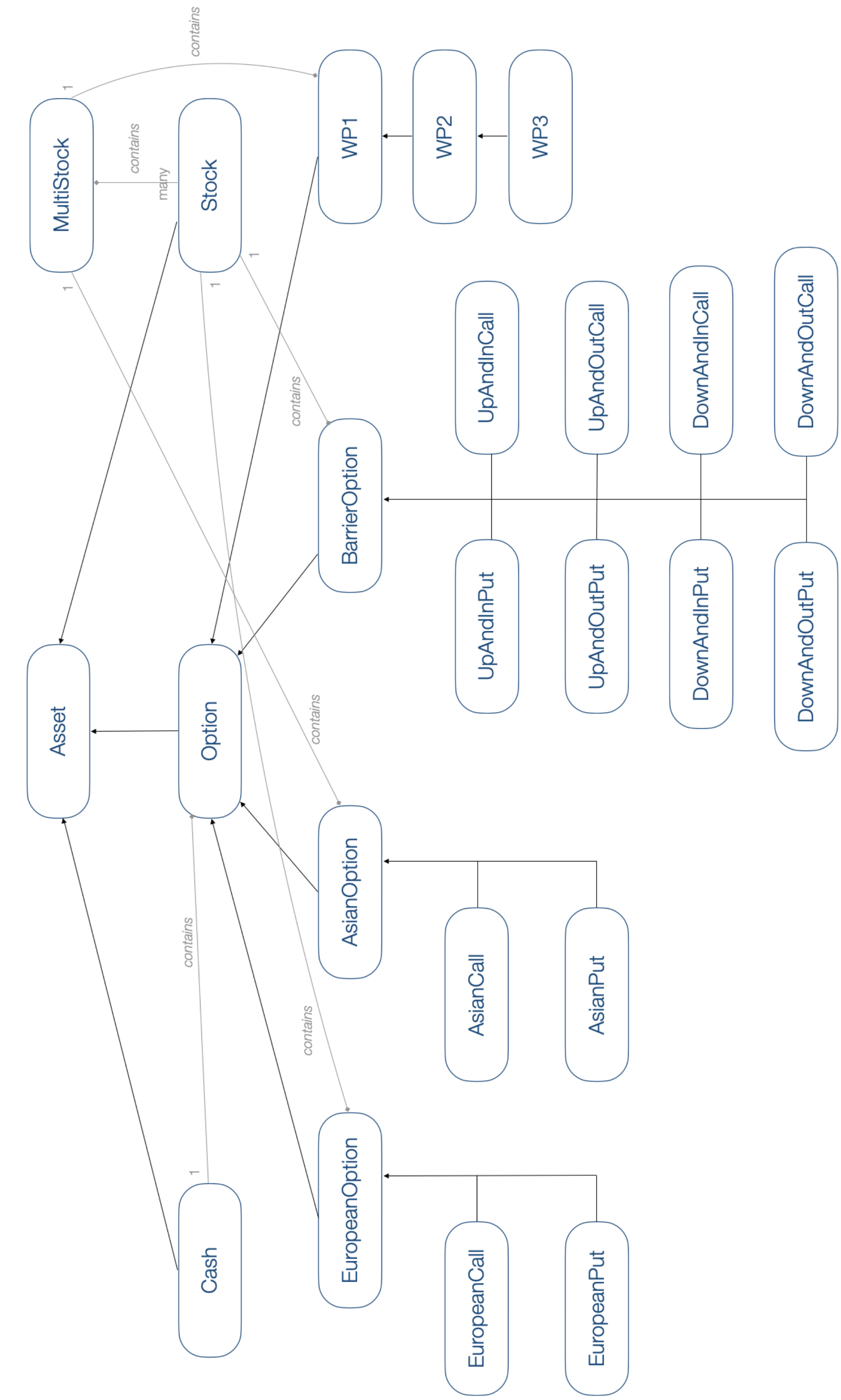


Figure 5.1: Market components' classes and their relationships.

The hedging horizon is naturally defined by the maturity of the derivative to be hedged. The rebalancing schedule is established through a branching factors vector, whose primary role will be explained in the context of scenario trees. However, the size of this vector sets the time interval dt and, consequently, the dates when portfolio adjustments are allowed. In particular, if β is the branching factors vector's length, then

$$dt = \frac{T}{\beta},$$

i.e. the hedging portfolio is constructed at time $t = 0$ and then potentially rebalanced at $t = n \cdot dt$, with $n = 1, ..., \beta - 1$. Indeed, the last rebalancing time is $T - 1 = (\beta - 1) \cdot dt$: the portfolio at maturity corresponds to the one set up in the previous time step. As an example, the tree in Figure 2.1 has a branching factors vector equal to [2,2,2] and $dt = \frac{T}{3}$.
According to the chosen number of replications, Monte Carlo simulations[1] are performed to generate real scenarios for testing the chosen hedging strategies. Starting from the given initial prices, the values of each market element are computed until the end of the hedging horizon:

- stock paths are simulated via Geometric Brownian Motion, leveraging the statistical properties estimated from real data;
- option values at each time step are determined according to their pricing models, i.e. the Black-Scholes-Merton model for vanilla European options, the corresponding analytical formulas for barrier options and the Monte Carlo-based risk-neutral valuation for both Asian options and worst performance derivatives[2]. The underlying assets values at each time step are those described in the previous point.

The framework is now established and ready to determine and apply an hedging strategy using stochastic optimization.

5.1 Reinforcement Learning Style

The hedging problem was approached following a Reinforcement Learning (RL) style framework. RL is based on the paradigm of trial and error, so an agent (the decision maker) learns to perform actions that lead to the highest rewards over time by interacting with an environment in order to maximize an expected cumulative

[1] See section 5.2 for a better understanding.

[2] Details on the derivatives employing Monte Carlo simulations are provided in subsection 5.2.1.

reward: the agent observes the current state of the environment, takes actions, receives rewards and updates its knowledge to improve future actions. Typically, RL methods do not rely on a precise mathematical model and instead employ heuristic techniques. In this case, however, a well-defined mathematical model is available. Specifically

1. the hedging agent observes the current state of the environment (the financial market), which consists of the present values of the hedging assets (`current_values`) and the current time (`current_time`);

2. based on the current state, the agent selects an action by solving the hedging problem outlined in Chapter 2. This process involves constructing a scenario tree from the current time to maturity by appropriately using the branching factor corresponding to the current step. From the solution of the hedging problem, only the quantities $x_j^{n_0} \quad \forall j \in \mathcal{A}$ (`current_quantities`) are considered, since they are those practically applicable at the current time;

3. after making the decision, the agent moves to a new environment state. However, unlike typical RL settings, the next state does not depend on the taken action. The environment's evolution follows a predefined sequence of states determined by the simulated financial market values, which was generated in advance (in Monte Carlo test scenarios, as explained in the previous section).

Unlike traditional RL, this approach does not rely on a reward function to drive decision making; instead, decisions follow a well defined model. For this reason, the approach only follows the style of reinforcement learning rather than fully leveraging its principles.
The implemented RL algorithm is presented below.

```python
for current_rep in range(env.reps): # Monte Carlo replications
    current_time, current_values = env.restart() # Restart the environment, by setting time to zero
    done = False
    # Loop in RL style: the agent observes the current state, takes an action and collects the correspondent reward before moving to the next state.
    while not done:
        if current_time != 0:
            # When time is not zero, save the stock prices history.
            initialize_past_values(env, current_rep, current_time)

```

```
        state = {'asset_values': current_values, 't': current_time} # state of the environment

        # Branching factors are progressively reduced in reverse order (excluding the last current_time stages)
        # to improve the performance when calling the hedgingSolver at further stages.
        if current_time == 0:
            bf = branching_factors
        else:
            branching_factors[:-current_time]

        # The agent takes an action by solving the optimization problem and rebalancing the holdings (saved in current_quantities)

        if isinstance(env.target_asset, BarrierOption):
            existsBarrierOption = env.target_asset.matrixExists[current_rep,current_time]
            current_quantities, reward, rebalancing_cost = stoch_agent.get_action(state, bf, existsBarrierOption)

        else:
            current_quantities, reward, rebalancing_cost = stoch_agent.get_action(state, bf)

        rebalancing_costs[current_rep, current_time] = rebalancing_cost

        # Save the rebalanced portfolio
        update_quantities(env, current_rep, current_time, current_quantities)

        # The environment step consists in getting aware of the next asset prices (already simulated through GetRealValues())
        current_time, current_values, done, info = env.step(current_rep, current_time)

    # At maturity the portfolio is not rebalanced, so the final holdings correspond to the ones of the previous time.
    # Cash:
    env.hedging_assets["Cash"].quantities[:, env.times-1] = env.hedging_assets["Cash"].quantities[:, env.times-2]
    # Stocks:
    for stock in env.hedging_assets["Multi_stock"].stock_list:
        stock.quantities[:, env.times-1] = stock.quantities[:, env.times-2]
    # Options:
    for option in env.hedging_assets["Options_list"]:
```

```
        option.quantities[:, env.times-1] = option.quantities[:, env.
    times-2]
```

The decision-making process slightly differs in the case of barrier options, but this will be better explained in section 5.4. Furthermore, rebalancing costs, if present, are recorded, e.g. in non-self-financing strategies or in those allowing for money withdrawals; if the initial wealth is not fixed, the cost of the portfolio construction is also included.
The `env.step()` function simply returns the values of the market components in the new state, corresponding to the previously computed market values for the subsequent time step.
The `done` flag is set to true after the last rebalancing decision at time $T-1$.
The `reward` returned by the `stoch_agent.get_action()` function[3], although not used in any way, trivially represents the value of the objective function of the hedging problem solved at that specific step.

5.2 Monte Carlo

Monte Carlo is a technique used to model and analyze problems that involve uncertainty. In more detail, it relies on repeated random sampling to estimate numerical results. Mathematically speaking, given a function h which depends on a random variable X with probability density function $p(x)$, the expected value of $h(X)$ can be estimated using the Monte Carlo approximation

$$\mathbb{E}[h(X)] = \int h(x)p(x)dx \approx \frac{1}{R}\sum_{r=1}^{R} h(X_r),$$

where $X_r \;\; r = 1, ..., R$ are independent samples of X and R is the number of Monte Carlo replications. The method is based on the Law of Large Numbers: as the number of trials R increases ($R \to \infty$), the empirical result tends to converge toward the expected theoretical value. Clearly, the estimation accuracy improves as the number of replications increases, but also computational cost has to be taken into account, since it may become a limiting factor.
Monte Carlo methods are particularly useful when analytical solutions are not easily attainable.
In this specific setting, Monte Carlo simulation is primarily used to estimate the behavior of the adopted hedging strategy. To achieve this, the latter is tested over a large number of replications R, referred to as *Monte Carlo replications*. The outcomes of all real-world simulated scenarios are then used to compute an estimate

[3] See section 5.3 for details.

of the average values of the performance metrics (hedging error, P&L), which will be described in Chapter 6. This approach also provides insight into the robustness of the implemented hedging strategies in different real-world situations.

5.2.1 Option valuation

Monte Carlo simulations can be used for option pricing, relying on the risk-neutral principle: paths are sampled, the expected payoff is computed in a risk-neutral world and then discounted at the risk-free rate to obtain the option value at a given time. The process for valuing an European-style Asian option with maturity T at time t is illustrated in the following scheme:

1. sample a random path for each underlying asset in a risk-neutral world;
2. calculate the derivative payoff; for an Asian option written on multiple underlying assets, the payoff is given by the average across assets of the time-average of each asset's price, as explained in section 4.3;
3. repeat steps 1. and 2. multiple times (e.g. 10^4) to obtain a large set of payoff samples for the derivative;
4. compute the mean of the total number of the sample payoffs: this is the estimate of the expected payoff in a risk-neutral world;
5. discount the estimate from T to t at the risk-free rate: the result represents an estimate of the value of the derivative at time t.

When an explicit pricing formula is not available, unlike vanilla or barrier options, the most effective method for option valuation is Monte Carlo simulation. This method proves to be effective, whether the payoff is based only on the final value of the underlying assets or influenced by their full price evolution. Thus, Monte Carlo is used in this thesis to evaluate path-dependent Asian options and worst-performance derivatives.
Practically speaking, starting from the theoretical result explained in section 3.1, to simulate the path followed by the underlying stock S_j, the hedging horizon is divided into β time intervals of length dt and the continuous equation (3.9) is discretized as follows:

$$S_j(t+dt) = S_j(t)\exp\left\{\left(\mu_j - \frac{\sigma_j^2}{2}\right)dt + \sigma_j Z_j\sqrt{dt}\right\},$$

where Z_j is a one-dimensional random sample from a multivariate (m-dimensional[4]) normal distribution, with correlation matrix $\boldsymbol{\rho}$.

[4] Recall that m is the considered number of stocks.

This allows the calculation of stock values at time $t + dt$ using their values at time t.
The following code represents the Monte Carlo implemented function for the valuation of European Asian options.

```
def MonteCarloPrice(self, S0, time_to_maturity, reps,
                    remaining_times, S_past=None):
'''
Compute the Monte Carlo price for the Asian option.
S0: Initial stocks prices
time_to_maturity: Time remaining until maturity
reps: Number of Monte Carlo simulations
remaining_times: Number of remaining time steps
S_past: Historical stock prices (None if current_time=0)
Return the average price across time steps and underlying assets;
this array is then used by AsianCall and AsianPut subclasses
to compute the final option price based on the payoff type (call or
    put).
'''

    # Extract underlying stocks, correlation matrix, and volatilities
    underlyings = self.multi_stock.stock_list
    rho = self.multi_stock.rho
    n_underlyings = len(underlyings)
    sigma = np.array([stock.sigma for stock in underlyings])
    mu = np.array([stock.mu for stock in underlyings])
    r = self.cash.risk_free_rate

    # dt = time step length
    dt = time_to_maturity / (remaining_times-1)

    # Initialize stock price matrices
    S = []
    for n in range(n_underlyings):
        S.append(np.zeros((reps, remaining_times)))
        S[n][:, 0] = np.repeat(S0[n], reps)

    # Simulate stock price paths
    for t in range(1, remaining_times):
        # In simulations settings (Geometric Brownian Motion):
        # S(t+dt) = S(t) * exp((r - 1/2*sigma**2) * dt + sigma * sqrt
    (dt) * Z)
        # where Z is a standard normal distribution

        # Generate correlated Brownian increments
        Inc = MultiStock.generate_BM_stock_increments(
            n_underlyings,
```

```
                r, mu, sigma, rho,
                dt, reps,
                self.rnd_state,
                risk_free = True)

        for n in range(n_underlyings):
            # Update stock prices using the geometric Brownian motion formula
            S[n][:, t] = S[n][:, t-1] * np.exp(Inc[n])  # S[n].shape = (reps, times)

    # If historical prices are provided, concatenate them with simulated prices
    if type(S_past) == np.ndarray:
        for n in range(n_underlyings):
            S_past_1asset = S_past[n,:].reshape(1,-1)   # (1, times_past)
            S_past_1asset = np.tile(S_past_1asset, (reps, 1))   # (reps, times_past)
            S[n] = np.hstack((S_past_1asset, S[n]))   # (reps, times_past+times)

    # Compute time averages for each simulation path
    time_means = np.zeros((n_underlyings, reps))
    for n in range(n_underlyings):
        time_means[n] = np.mean(S[n], axis=1)

    # Compute the average of the means across all underlying assets
    asset_means = np.mean(time_means, axis=0)  #(reps)

    payoffs = self.payoff_formula(asset_means)
    price = np.exp(-r * time_to_maturity) * np.mean(payoffs) # Discounted average payoff

    return price
```

5.3 Scenario trees: stochastic models and arbitrages

The Geometric Brownian motion approach discussed in section 3.1 focuses only on generating price paths. Whereas MM simultaneously models prices and node probabilities, the GBM approach requires the probabilities to be computed separately. Two alternatives are proposed, which involve formulating a Moment Matching problem to calculate the probabilities π_k of the children nodes $k \in [K]$, while maintaining the characteristic moments for the log-returns recalling (3.8), i.e.:

- first moment of log-returns of stock j: $\left(\mu_j - \frac{\sigma_j^2}{2}\right) dt$;
- second moment of log-returns of stock j: $\sigma_j^2 dt + \left[\left(\mu_j - \frac{\sigma_j^2}{2}\right) dt\right]^2$;
- the expectation of the product between log-returns of two different stocks j and l: $\sigma_j \sigma_l dt \cdot \rho_{j,l} + \left(\mu_j - \frac{\sigma_j^2}{2}\right) dt \cdot \left(\mu_l - \frac{\sigma_l^2}{2}\right) dt$.

The historical statistical properties $\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\rho}$ are those estimated from a set of historical data, as introduced at the beginning of this chapter.
The previously listed properties of log-returns are matched with those obtained from one step of the scenario tree, as explained in section 3.2. Note that in this specific case, only probabilities are decision variables, since stock prices have already been simulated through GBM. Two Moment Matching formulations have been implemented to determine the children probabilities π_k starting from their parent node:

1. one variant (`BrownianMotionForHedging_Gurobi`) involves the minimization of the squared distance between the first and second moments and the expectations of the product between log-returns of two different stocks (to capture correlation); the objective function is the sum of the squared difference between each expected property and its counterpart resulting from the tree. This model is addressed with the Gurobi Optimizer, since it is a quadratic programming problem. Indeed, since scenario tree's statistical properties are computed as follows

$$\text{first moment: } \sum_{k \in [K]} \pi_k \ln \left(\frac{S_j^k}{S_j^{a(k)}} \right)$$

$$\text{second moment: } \sum_{k \in [K]} \pi_k \left[\ln \left(\frac{S_j^k}{S_j^{a(k)}} \right) \right]^2$$

$$\text{expectation of the product: } \sum_{k \in [K]} \pi_k \left[\ln \left(\frac{S_j^k}{S_j^{a(k)}} \right) \right] \left[\ln \left(\frac{S_l^k}{S_l^{a(k)}} \right) \right],$$

   and considering that the squared distance is minimized, the objective function is quadratic with respect to the decision variables π_k.

2. the second approach (`BrownianMotionForHedging`) considers the first and second moments and the correlations, which however introduces non linearity. Thus, this Moment Matching minimization problem is solved using SLSQP

(Sequential Least Squares Programming), since it is suited for non linear optimization problems.

MM stochastic model (`MomentMatchingForHedging`) for the generation of stock prices follows exactly what was described in section 3.2, considering the log-returns statistical properties (3.16), (3.17), (3.18) and (3.19) which have to be matched with the historical estimated mean, standard deviation, correlation and skewness.

The stochastic models described above are used by the `ScenarioTree` class, which is responsible for the construction of scenario trees. One of the three available models can be chosen, each offering to `ScenarioTree` a function to simulate one time step (`simulate_one_time_step()`), whose purpose is to generate children nodes from their parent node. The connection between stochastic models and the scenario tree's class is depicted in Figure 5.2 and in the following function:

```
    def _generate_one_time_step(self, n_scenarios, parent_node):
    '''Given a parent node and the number of children to generate, it returns the children with corresponding probabilities'''
        prob, obs = self.stoch_model.simulate_one_time_step(
            parent_node=parent_node,
            n_children=n_scenarios
        )
        return prob, obs
```

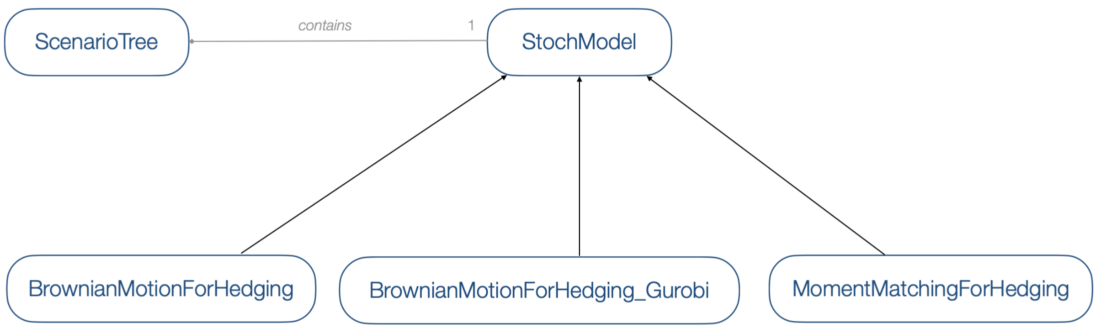


Figure 5.2: Relationship between scenario tree and stochastic models classes.

Each `simulate_one_time_step()` function has to:

1. simulate children stock prices through the chosen stochastic model;
2. assure that the generated values are arbitrage free;
3. compute the children nodes probabilities;

4. find the new values for the remaining hedging assets: Black-Scholes-Merton model is employed for vanilla options, while cash follows a deterministic risk-free dynamics.

To avoid arbitrage opportunities, both stochastic models are combined with one approach from section 3.3. However, both methods have drawbacks. The absence of dominant strategies is verified by solving problem (3.26) − (3.29) to ensure consistency in the computed values, but without generating or modifying them. If the check fails, new stock values are generated and the check is repeated. As a result, a `while` loop is required, along with a proper maximum number of iterations before concluding that arbitrage-free prices cannot be found in that specific situation.
On the other hand, adjusting the already simulated prices according to problem (3.33) − (3.31) requires a careful tuning of the α parameter, which is not a trivial task.

The `ScenarioTree` class is then employed by the agent. More precisely, in the `get_action()` function it generates a scenario tree starting from the current state until maturity, which is then used to solve the hedging problem (discussed in Chapter 2) by the `HegingSolver` class[5].
Figure 5.3 shows a schematic representation of how the different classes interact to compute the optimal portfolio rebalancing decision at each step.

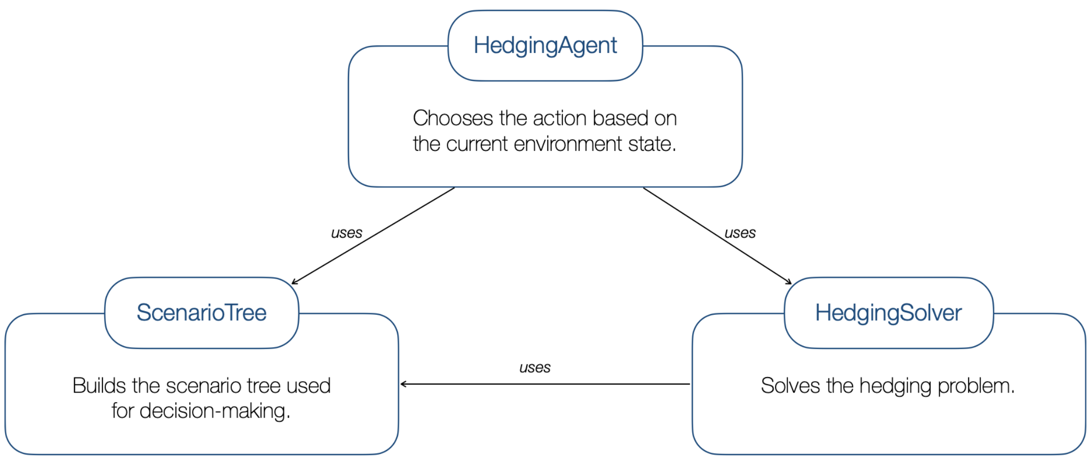


Figure 5.3: Hedging classes scheme.

[5] Also a `SuperReplicationHedgingSolver` class is implemented, related to problem described in section 2.4.

At this point, the agent `get_action()` function can be presented in more detail.

```
def get_action(self, state, branching_factors, exists = None):

    '''Determine the hedging action based on the current market state. The state is a dictionary containing 'asset_values' and current time 't'. Based on the current asset values, the agent rebalances the portfolio, keeping in mind the long-term objective of hedging the position in the target asset.'''

        asset_values = np.array(state['asset_values']) # current values of all assets
        t = state['t'] # current time step
        n_assets = len(asset_values) # number of assets in the environment

        # Initialize a ScenarioTree to simulate future scenarios for hedging
        MyTree = ScenarioTree(
            name='HedgingTree',
            branching_factors=branching_factors,
            len_vector=n_assets, # number of assets
            initial_value=asset_values, # initial_value is the initial price of all the assets in the market.
            stoch_model=self.env.stochastic_model # Stochastic model used to generate scenarios
            )

        # Find the optimal hedging strategy.
        # The HedgingSolver object will store into each hedging assets' initial_quantity attribute its holding after the rebalancing decision
        if self.super_replication:
            MyHedging = SuperReplicationHedgingSolver(
                    # scenario tree used for simulations
                    Tree = MyTree,
                    # target asset to hedge
                    target_asset = self.env.target_asset,
                    # assets used for hedging
                    hedging_assets = self.env.hedging_assets,
                    exists = exists, # useful only for Barrier Options
                    self_financing = self.self_financing,
                    withdraw_possibility = self.withdraw_possibility,
                    liquidity_fund = self.liquidity_fund,
                    current_rebalancing_time = t
                    )
```

```
        else: # no super-replication, the default hedging problem is
    solved
            MyHedging = HedgingSolver(
                    # scenario tree used for simulations
                    Tree = MyTree,
                    # target asset to hedge
                    target_asset = self.env.target_asset,
                    # assets used for hedging
                    hedging_assets =
                    self.env.hedging_assets,
                    exists = exists, # useful only for Barrier
    Options
                    gamma = self.gamma,
                    self_financing = self.self_financing,
                    withdraw_possibility = self.withdraw_possibility,
                    current_rebalancing_time = t
                    )

        # Initialize the list to store the quantities (holdings) of
    each hedging asset after the rebalancing decision.
        current_quantity = []

        # Cash:
        current_quantity.append(
            self.env.hedging_assets["Cash"].initial_quantity)

        # Stocks:
        for stock in self.env.hedging_assets["Multi_stock"].
    stock_list:
            current_quantity.append(stock.initial_quantity)

        # Options:
        for option in self.env.hedging_assets["Options_list"]:
            current_quantity.append(option.initial_quantity)

        # Objective value of the optimization problem solved
        reward = MyHedging.M.ObjVal

        rebalancing_cost = MyHedging.rebalancing_cost

        # return the rebalanced portfolio (current_quantity)
        # and the objective value of the HedgingSolver
        return current_quantity, reward, rebalancing_cost
```

5.4 Barrier options

Unlike traditional options, the barrier ones present a more complex structure, since it is necessary to determine if they are active by checking the barrier. This check is not continuously made, but it takes place only at the rebalancing dates, when the current underlying price $S(t)$ is observed and compared to the barrier H. Consequently:

- up-and-in options come into existence in the first time step when $S(t) \leq H$ is verified, while up-and-out cease to exists in the same situation;

- down-and-out options are deactivated when $S(t) \geq H$ occurs for the first time; down-and-in are instead activated under the same condition.

To handle this structure, a matrix (`matrixExists`) is used to track step by step the existence of the barrier option. It is initialized as `False` for in-options and `True` for out-options. Then, through the `hitTheBarrier` function, this matrix is updated the first time the barrier event is triggered. As an example, the implemented code for up-and-in options is presented below.

```
def initializeExists(self, n_rep, n_times): # exists only if the barrier is hit
        self.exists = [False for i in range(n_rep)]
        self.matrixExists = np.full((n_rep, n_times), False)

def hitTheBarrier(self, current_underlying_price, current_rep, current_time):

    if (current_underlying_price >= self.barrier):
        self.exists[current_rep] = True #the option comes into existence
        self.matrixExists[current_rep, current_time: ] = True
```

The boolean list called `exists` is useful to compute the final payoff for the simulated real-world scenarios: each entry could become `True` when the barrier is hit for the first time, resulting in a specific payoff formula. Otherwise, if the up-and-in option is never activated (i.e. at the end of the hedging horizon the correspondent entry of `exists` is still `False`), the payoff will be zero.
From the hedging problem perspective, the agent must know whether the option exists in the current state to make a proper decision for the hedging of the derivative at maturity. This is why in the RL-style formulation the code slightly differs for barrier options, allowing for an additional input in the `get_action()` function that informs the agent about the current existence state of the option.

```
if isinstance(env.target_asset, BarrierOption):
    existsBarrierOption = env.target_asset.matrixExists[current_rep, current_time]
    current_quantities, reward, rebalancing_cost = stoch_agent.get_action(state, bf, existsBarrierOption)
```

Then, this information will be passed to the hedging solver. At this point, the payoff that the scenario tree is supposed to replicate is determined as follows:

- for knock-out options: if the option does not exist anymore at the current decisional step, the payoff will certainly be 0. However, if the option still exists in real world, the barrier event is checked for the future values simulated by the tree. If the barrier is hit in a subsequent simulated step, the payoff will be zero. Otherwise, if the option continues to exist until maturity, the payoff will follow a specific formula depending on the type of option (call or put);
- for knock-in options: if the derivative already exists at the time of making the decision, the payoff is given by the corresponding formula (depending on whether it is a call or put option). If the option has not come into existence yet, the barrier event has to be checked for the future values simulated by the tree. If the barrier is hit in any of the future steps, the payoff follows the formula earlier mentioned. In contrast, if the barrier is never reached until maturity, the option remains inactive and its payoff is zero.

5.5 WP derivatives

5.5.1 Management of coupons: aggregation function

The critical aspect of worst-performance derivatives is that they involve the periodic payment of coupons, which represent liabilities for the bank over the investment horizon. Thus, the hedging strategy must be able to sustain also these cash outflows. Therefore, the hedging strategy should aim to replicate the maturity payoff while guaranteeing sufficient funds to meet coupon obligations. Obviously, it is unsustainable to have as many rebalancing dates as periodic payment dates: each portfolio adjustment is related to transaction costs and if they are too frequent, the hedging strategy would require an excessive amount of capital to cover the target derivative, resulting in losses rather than profits (or, at least, a zero P&L). Instead, a carefully chosen rebalancing schedule aims to achieve a trade-off between maintaining a good performance of the hedging and controlling costs, ensuring that the strategy remains practical and robust over the investment horizon. For this reason, an aggregation function has been introduced, allowing the rebalancing at time t to incorporate all coupons scheduled for payment between the previous rebalancing date $t-1$ and t. In the simplest case where the coupons are deterministic, as in the WP1 derivative, the aggregation results in

$$l_t = \sum_{k=t-1}^{t} C_k,$$

i.e. the liability that can be potentially sustained at time t is the sum of the coupons whose payment dates are between $t-1$ and t.
A more complex situation arises in the WP2 derivative, characterized by stochastic coupon payments. Although the coupons are aggregated between rebalancing dates, they are paid only if, at the rebalancing date, the worst-performing underlying asset S_{w_t} has a value higher than its digital barrier. Recalling (4.9), this means

$$l_t = \sum_{t_j \in (t-1;t]} C_{t_j} = \begin{cases} \sum_{t_j \in (t-1;t]} G & \text{if } S_{w_t}(t) \geq g \cdot S_{w_t}(0) \\ 0 & \text{otherwise} \end{cases}.$$

Note that the digital barrier is checked only at the evaluation time t when coupons are aggregated and not at every coupon date as stipulated by the derivative's contractual terms.
WP3 derivative shares a similar structure: the periodic coupons are aggregated, but the general liability term l_t also accounts for not paid past coupons and, potentially, the early redemption amount.
When an instance of these derivatives is created, the periodic coupons and their respective payments dates must be provided. Given the number of time steps in the hedging horizon, the coupons are immediately aggregated at these specific points in time, allowing them to be used in both the real-world simulation process and the hedging problem.
Liability management in scenario trees is not a straightforward task. Each node, depending on its corresponding time step, has a different liability. The following code shows the implemented algorithm to associate the right liability to each non-leaf node in the case of WP1.

```
threshold = []
t = 1
for j in range(len(self.Tree.branching_factors)):
    t*=self.Tree.branching_factors[j]
    threshold.append(t)
for j in range(len(threshold)):
    threshold[j] = threshold[j]+ sum(threshold[:j])
for i in self.non_leaf_nodes[1:]: # for each non leaf node
    for j in range(len(self.Tree.branching_factors)-1):
        if i <= threshold[0]:
```

```
            liability_term = self.liabilities[1]
        elif threshold[j-1] < i <= threshold[j]:
            liability_term = self.liabilities[j+1]
```

First of all, the number of nodes that are related to the same time step has to be found and stored in `threshold`. For example, with a branching factors vector [5,4,3], `threshold` will be equal to $[5, 20, 60]$, meaning that the first 5 nodes are those at the first time step, the following 20 are related to the second time step, while the last 60 are leaf nodes. Then, the same list is updated in order to contain the thresholds for the nodes' indexes, e.g., in the same example, it becomes [5,25,85]. This means that non-leaf and non-root nodes with index between 1 and 5 are related to the first step, those with index between 6 and 25 are modelling the second time step, and so on. Based on this structure, the correct liability is assigned to each tree node.
The situation becomes even more complex for WP3, as it also requires tracking unpaid past coupons and verifying the occurrence of an early redemption event. The next two sections provide a step-by-step explanation of the WP3 implementation, distinguishing between what is done in the scenario tree hedging problem and what in real-world simulation. The implementation of WP2, and even more evidently the one of WP1, are simplified cases of the following.

WP3: Scenario tree simulation

1. Coupons and early redemption are treated as liabilities in the current time t, specifically in cash balance constraints as discussed in Chapter 2. In each tree node:
   (a) verify if early redemption has not already occurred along the path until the current node[6]:
      - if ER has already taken place, the liability in the current node is zero
      - otherwise, control if the coupon has to be paid in the current node by checking the digital barrier of the current worst-performer underlying
        - if yes, the current liability is given by the aggregated coupons (following what explained in subsection 5.5.1) at the specific current time step; additionally, all past coupons that have never been paid are now considered;
        - if no, no coupons of any kind are paid in the current node.

   (b) if early redemption occurs at the current node, the early redemption amount is added to the current liability term.

[6] This path is made by real observed prices from time 0 to the current time (not included).

2. The maturity payoff to be considered for replication by the tree is given by (4.11), i.e. in each leaf node:
    - if ER has occurred along the path[7], the payoff is equal to zero;
    - otherwise, the worst-performer underlying at maturity determines a non-zero payoff, following (4.7).

WP3: Real-world simulation

1. From a pricing perspective, the derivative's value at time $t = 0$ is equal to the expected discounted payoff, coherently with risk-neutral valuation. In this case, the "payoff" includes all cash flows realized up to and including maturity (brought forward in time to T), rather than being limited to the final maturity payment. These include the aggregated coupons (accounting for the memory effect), the potential early redemption amount and the possible non-zero cash flow at maturity, linked to the worst-performer at T;
2. the derivative's values at $t = 1, ..., T - 1$ are obtained through the risk-neutral valuation formula, unless early redemption has taken place at an earlier point in the real-world scenario. In that case, the derivative's value is zero, reflecting the fact that it ceases to exist after the early redemption.
3. the derivative's value at maturity is given by its payoff, i.e. the actual cash flow which takes place at maturity; this is coherent with what is considered as payoff to be replicated in the hedging problem solved in scenario trees.

[7] The complete path (from $t = 0$ to maturity) on which the check is performed consists entirely of values simulated by the scenario tree in the tree built at time 0. For the trees constructed at subsequent time steps, the path is composed by the prices actually observed in real world up to the current time, while the prices from the current time to maturity are those simulated by the scenario tree.

Chapter 6

Analysis

Before delving into the description of the conducted analyses, it is important to clarify the adopted performance metrics: hedging error and profit and loss.
Hedging error is, by definition, the difference at maturity between

- the value of the hedging portfolio lastly rebalanced at $T-1$ and kept until maturity T, i.e.
$$V_P^T(T) = \sum_{j \in \mathcal{A}} x_{j,T-1} p_{j,T}.$$
Since the risky hedging instruments are assumed to have a maturity that coincides with the one of the target derivative, their value at maturity is in fact equal to their payoff, i.e. $p_{j,T} = \Psi_j(T)$;
- the target asset cash flow at maturity, i.e. its payoff $\Psi(T)$.

Mathematically,
$$\text{he} = V_P^T(T) - \Psi(T). \tag{6.1}$$
For an efficient hedging strategy, this value should be close to zero.
Profit and loss, instead, measures the net financial outcome of a strategy, taking into account any cash flows C_t generated throughout the holding period. Formally,
$$\text{P\&L} = \sum_{t=0}^{T} C_t \cdot \mathrm{e}^{r(T-t)}.$$
Note that all P&L components are brought to time T. For the unhedged position, this is simply
$$\text{P\&L}_{unhedged} = \Pi(0)\mathrm{e}^{rT} - \Psi(T) - \sum_{t=0}^{T} l_t \cdot \mathrm{e}^{r(T-t)},$$

i.e. the difference between the cash inflow at time $t = 0$ (the target asset premium $\Pi(0)$) and the cash outflows, divided between the payoff at maturity $\Psi(T)$ and the sum of all potential liabilities l_t incurred during the hedging horizon.
Considering a hedging strategy, its payoff must additionally reflect the presence of a hedging portfolio, which is assumed to be sold at maturity, resulting in a positive cash flow equal to $V_P^T(T)$. Thus:

$$\text{P\&L}_{hedged} = (\Pi(0) - W_0)\mathrm{e}^{rT} - \Psi(T) + V_P^T(T) + \sum_{t=1}^{T-1} q_t \cdot \mathrm{e}^{r(T-t)}, \tag{6.2}$$

where W_0 is the portfolio's initial cost and q_t represents the additional cash flow occurred at time t, which could be either positive or negative. Assuming that the hedging portfolio fully covers the additional liabilities l_t, they are neutralized and do not appear in the final formula.
If the hedging strategy adopted is self-financing (i.e. there are no negative q_t), without withdrawal possibility (positive q_t are absent, too) and with an initial cost equal to the target asset price, its profit and loss (6.2) coincides with the expression of the hedging error (6.1).

6.1 Hedging statistics

This section provides a preliminary evaluation of the stochastic hedging strategy's performance in terms of hedging error and profit and loss. In order to show the effectiveness of the proposed algorithm, the classical hedging problem (2.1) − (2.9) is applied to hedge European vanilla and Asian options. The simulation setting involves $n = 5000$ Monte Carlo replications and a branching factors vector [25,3,3], meaning that there are 4 time instants and 3 rebalancing dates.
Generally speaking, the underlying stocks available in the considered market are ENEL.MI, MMM, TSLA whose initial prices are, respectively, 102.05, 242.92 and 108.19 \$. The historical stock parameters $\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\rho}, \boldsymbol{\varsigma}$ are estimated from a set of historical data from 05/01/2022 to 05/01/2024. Moreover, in all the following analyses, the risk free rate is set to be $r = 0.0398$.
Risky assets present transaction costs that are equal to 1 % of their value.
A self-financing strategy without possibility to withdraw money along the hedging horizon is considered in this first analysis: all periodic cash flows are fully covered by the hedging portfolio itself, without any cash inflow or outflow. The only allowed cash flow occurs at time zero and is used for portfolio construction. In this specific case, it is set exactly equal to the price of the target asset. As a result, the hedging error coincides precisely with the P&L.
Regarding the guarantee that scenario trees do not contain arbitrage opportunities, the choice between the two methods described in section 3.3 was not based on any

specific reason. Preliminary tests showed that both methods successfully prevent arbitrage, with comparable computational demands. One approach was more efficient in some cases, while the other excelled in others, meaning that there is not a clear rule to justify the selection of one over the other.

6.1.1 Hedging vanilla European options

With a European vanilla option as target asset, the hedging is performed with only the underlying stock and the bank account as hedging assets. The hedging strategy is analyzed through two vanilla European style options: one call option with underlying ENEL.MI and strike price $K = 50$ and one put option with underlying TSLA and strike price $K = 300$. Both options are set to mature in one year (i.e. $T = 1$). Results demonstrates that the adopted strategy is able to efficiently hedge the short position in both vanilla options: as shown in Figure 6.1 and Figure 6.2, the profit and loss empirical distribution over the 5000 Monte Carlo replications exhibits significantly lower variance with respect to the unhedged position, resulting in a highly concentrated distribution around 0.

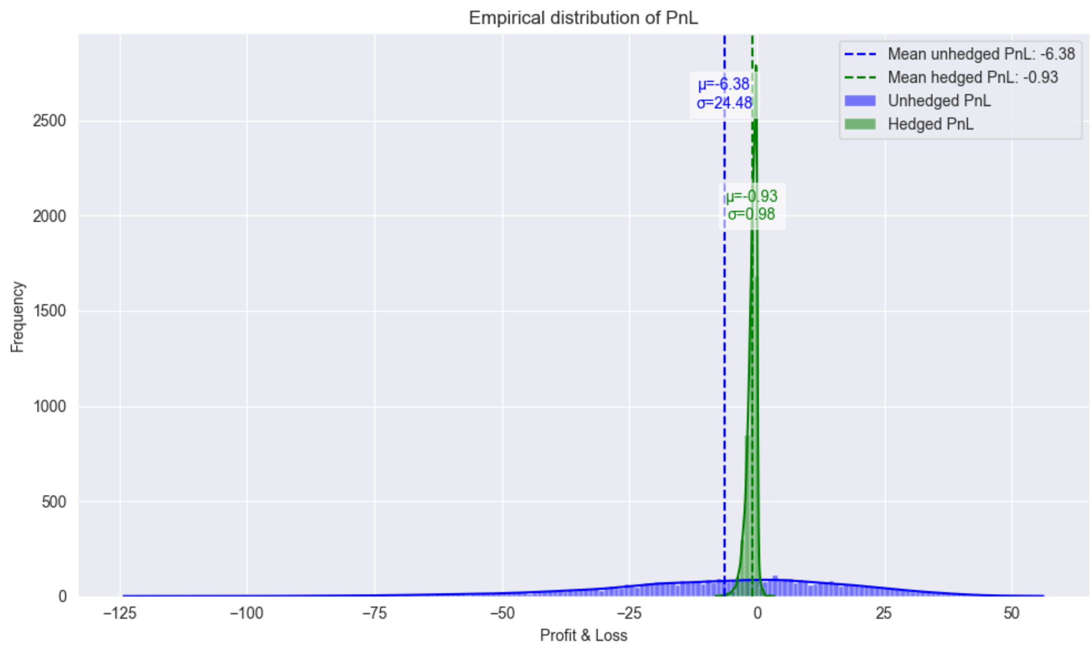


Figure 6.1: P&L empirical distributions for a European call option: the unhedged (blue) one is compared to the hedged (green) one.

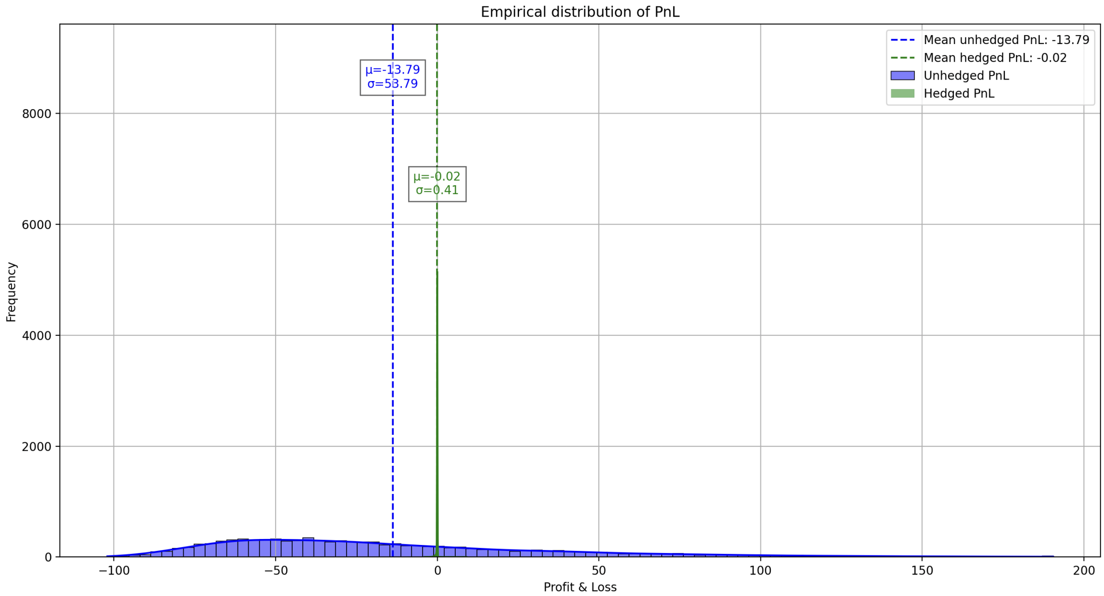


Figure 6.2: P&L empirical distributions for a European put option: the unhedged (blue) one is compared to the hedged (green) one.

6.1.2 Hedging Asian options

The performance is now evaluated for Asian options, in particular for one put with strike price $K = 250$ and one call with strike $K = 50$, both with maturity $T = 1$ year. A more diversified hedging portfolio is chosen due to their complex payoff structure, in contrast to vanilla options. First of all, unlike vanilla options, they are based on a set of underlyings, which are all included as hedging instruments. All three stocks available in the market are considered and, additionally, one put and one call option for each stock, whose characteristics are summarized in Table 6.1. Strategies for both Asian options also include the cash position in the hedging assets.

Empirical P&L distributions obtained from the simulated Monte Carlo scenarios are depicted in Figure 6.3 and Figure 6.4. Also these cases confirm the effectiveness of the adopted hedging strategy. Similarly to the results obtained for vanilla options, this is evidenced by the lower variance in the P&L with respect to the variance in the unhedged positions. Also, the hedging error is closer to 0, meaning that at maturity the hedging portfolio is able to almost perfectly replicate the target asset's payoff.

Target asset	Vanilla options	Underlyings ENEL.MI	MMM	TSLA
Asian call	European Put	130	260	120
	European Call	70	200	90
Asian put	European put	170	300	200
	European all	100	200	90

Table 6.1: Strike for vanilla options used as hedging instruments.

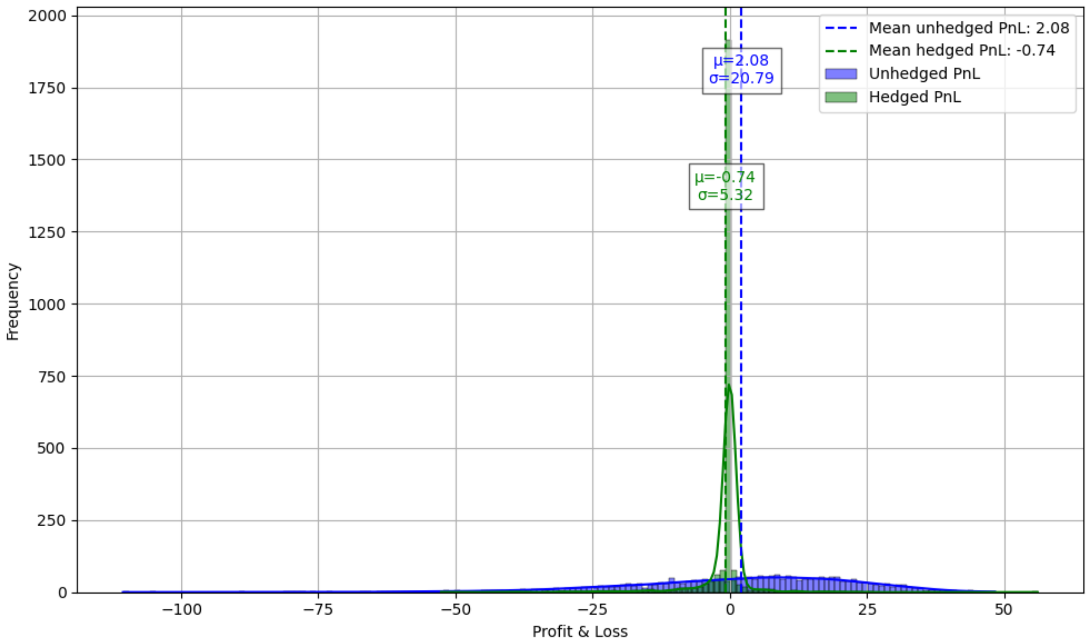


Figure 6.3: P&L empirical distributions for an Asian call option: the unhedged (blue) one is compared to the hedged (green) one.

6.1.3 Comparative analysis: European vanilla vs Asian options

To provide a clearer overview of all the results discussed so far, Table 6.2 summarizes the significant performance metrics. The statistical evidence is clear: the implemented stochastic optimization algorithm is able to find very effective hedging strategies for both European and Asian options. Profit and loss means are close to zero in all cases, while standard deviations slightly differ. In particular, exotic

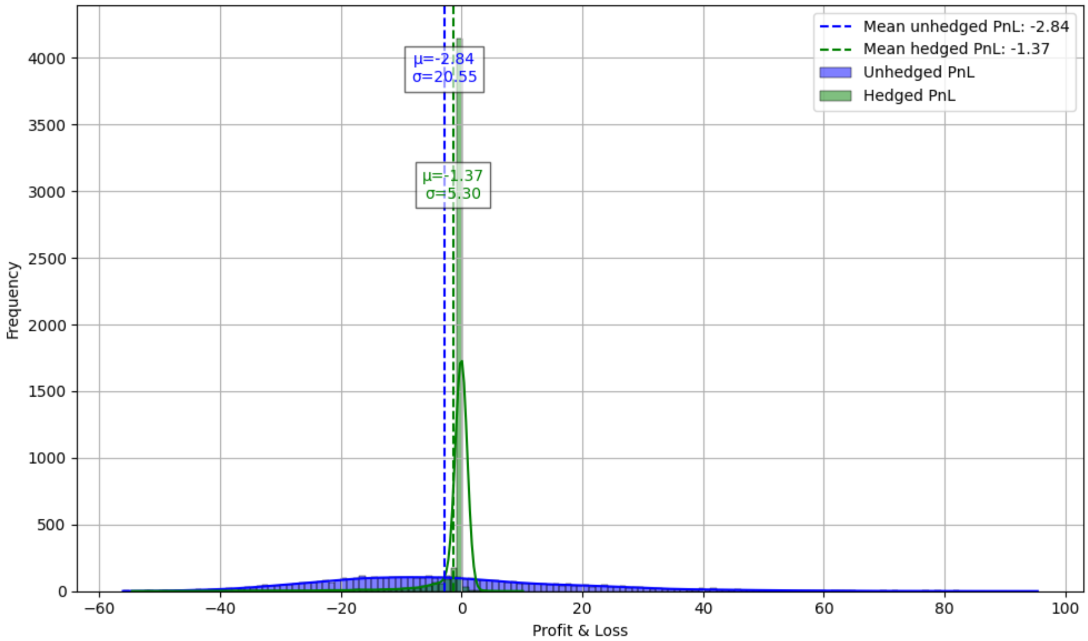


Figure 6.4: P&L empirical distributions for an Asian put option: the unhedged (blue) one is compared to the hedged (green) one.

Target asset	P&L hedged		P&L unhedged	Time
	μ	σ	σ	
Vanilla call	0.93	0.98	24.48	1232.20 s
Vanilla 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 6.2: Statistical comparison of performance metrics over 5000 Monte Carlo simulations.

options exhibit higher values than vanilla ones.
This comparative analysis also reveals differences in the computational time required to perform the Monte Carlo simulation over 5000 replications, highlighting that exotic options are more computational demanding. Nevertheless, for Asian options, both higher standard deviation and computational time may be linked to the larger set of hedging instruments (for a total of 10 assets, compared to the

only 2 in vanilla cases), which inevitably causes an increasing in the complexity of the optimization process.

Until now, the evaluation of performance metrics has been limited to maturity, since it is the actual moment when it is important to achieve hedging due to the possible exercise of the option by the holder. This approach is justified by the fact that the optimization problems solved at each step are designed to minimize the replication error specifically at maturity. Nevertheless, it can be interesting to examine the hedging strategy's behaviour over the entire time horizon by tracking the target asset (which corresponds to the unhedged position), the replication portfolio and the resulting hedged position. For this purpose, step-by-step replication plots for the previously analyzed derivatives are provided. In particular, they show

- a blue line, which corresponds to the hedging portfolio values at each time step, computed as

$$V_P^t(t) = \begin{cases} \sum_{j \in \mathcal{A}} x_{j,t} p_{j,t} & t = 0, ..., T-1 \\ \sum_{j \in \mathcal{A}} x_{j,t-1} p_{j,t} & t = T \end{cases} ,$$

  i.e. the value $V_P^t(t)$ of the hedging portfolio at time t after eventual rebalancing at time t is given by the sum of all instruments $j \in \mathcal{A}$ values with holdings $x_{j,t}$ and values $p_{j,t}$. Note that risky assets' value at maturity T is equal to their payoff, i.e. $p_{j,T} = \Psi_j(T)$;

- a red line, representing the unhedged position, i.e. simply the value of the target asset. Its price is considered from $t = 0$ to $t = T - 1$, while at maturity its payoff is provided instead; since the taken position in the target asset is a short one, the unhedged values are changed in sign;

- a green line for the hedged position, which results from the sum of the previous two lines.

As illustrated in Figure 6.5, the hedging portfolio is able to accurately replicates the European option value, ensuring that the hedged position remains near to zero during the entire hedging horizon.
An Asian option case is depicted in Figure 6.6. While the objective of hedging at maturity is fully achieved (as already shown by the statistics at maturity, i.e., the P&L distribution characteristics), some intermediate stages may show less accurate replication, resulting in a hedged position not exactly equal to zero.

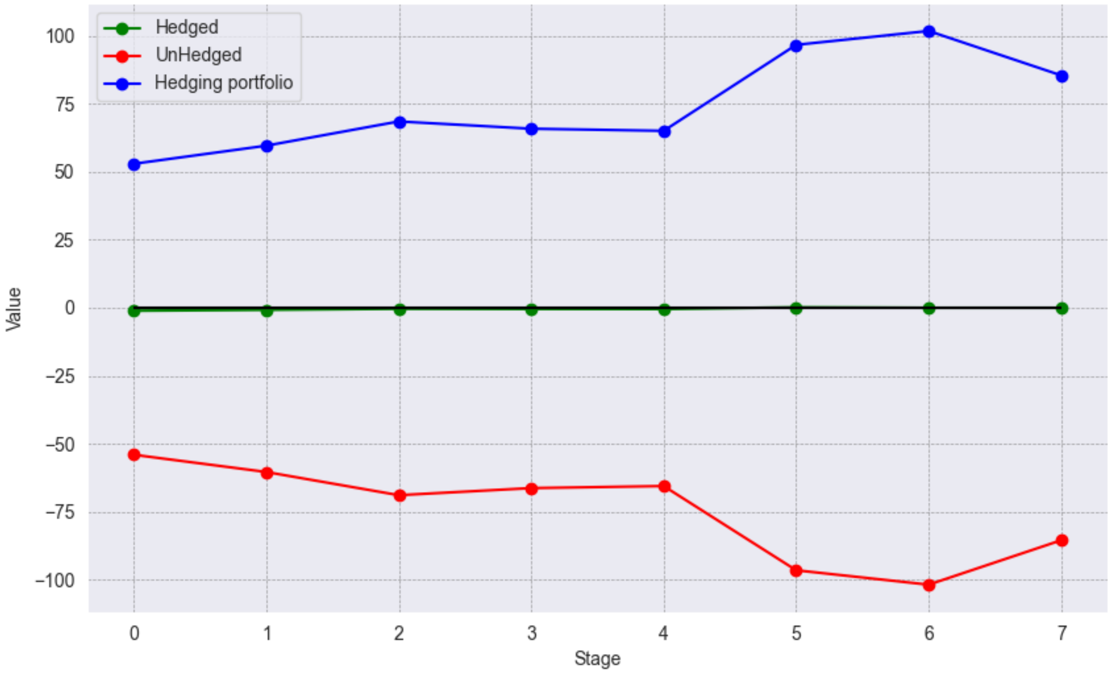


Figure 6.5: Step-by-step evolution of the hedging strategy for a Monte Carlo replication for a European call option.

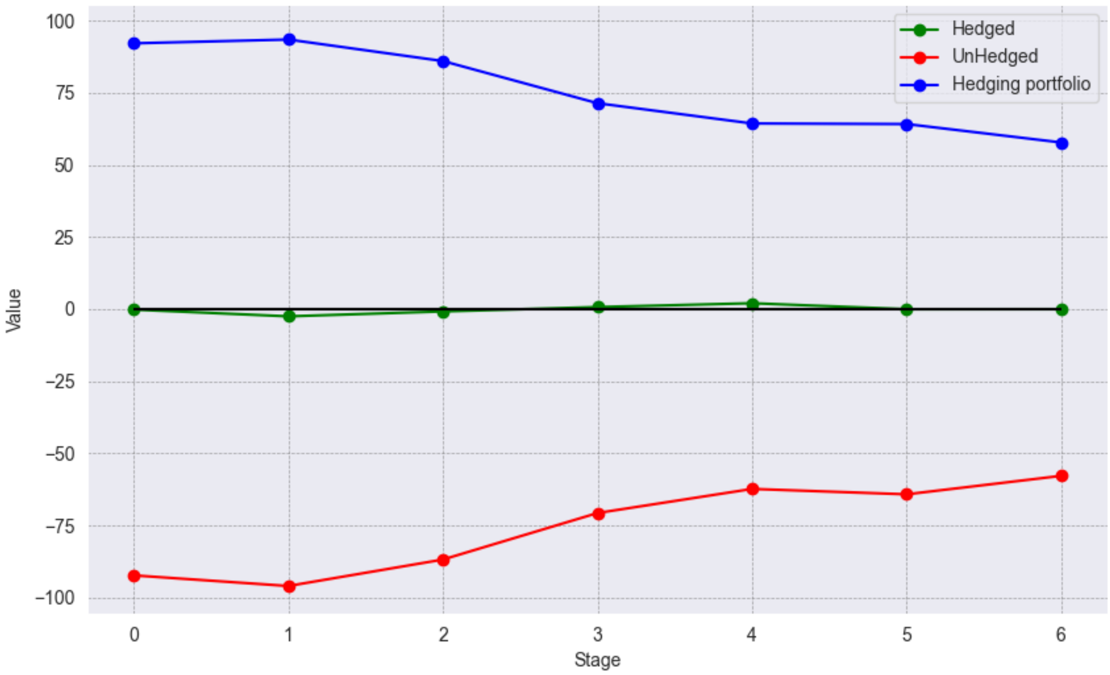


Figure 6.6: Step-by-step evolution of the hedging strategy for a Monte Carlo replication for an Asian put option.

Since Asian options are more complex and depend on multiple underlyings and thus they are subject to a large number of risk factors, the self-financing hedging portfolio may encounter some difficulties, despite the rebalancing. In a different Monte Carlo replication, shown in Figure 6.7, the gap between the target asset and the hedging portfolio values becomes more significant. However, even in this scenario of not perfect replication along the hedging horizon, the only liability the bank may face, i.e. the one at maturity, is fully covered by the value of the hedging portfolio at maturity. It is crucial to remember that the fundamental purpose of the hedging strategy is to ensure that the bank does not incur substantial losses at maturity. In the derivatives considered so far, no additional intermediate cash outflows are required. Thus, an imperfect step-by-step replication does not imply a worse performance at maturity, which remains unaffected by the absence of state-by-state replication. Nevertheless, achieving a good intermediate replication is a positive indicator of the hedging strategy's progress.

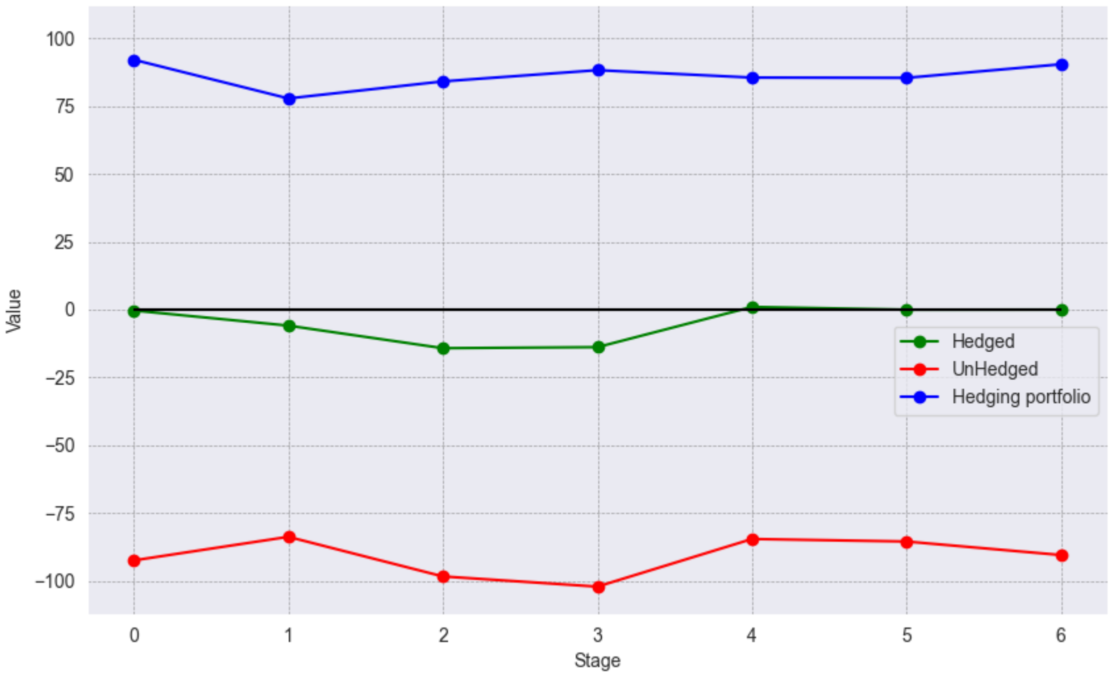


Figure 6.7: Step-by-step evolution of the hedging strategy for a different Monte Carlo replication for an Asian put option.

6.2 Branching factors sensitivity

By examining the relationship between branching factors and performance metrics, it is possible to have an insight into how different configurations affect both computational demands and solution quality. Table 6.3 presents the conducted sensitivity analysis of various branching factors (bf) with respect to the number of nodes, computation time, hedging error, cost of hedging and profit and loss for the hedged position. All quantities reported in the table are averaged over 1000 Monte Carlo replications. To provide a comparison with the unhedged position, the unhedged

bf	Nodes	Time	he		Cost		P&L	
			μ	σ	μ	σ	μ	σ
[10,5,4]	260	111.6	−1.688	16.00	93.41	2.06	−2.775	16.08
[25,6,5]	925	804.4	−0.623	5.67	93.22	0.50	−1.511	5.70
[25,15]	400	74.42	0.071	5.54	93.89	0.56	−1.516	5.56
[3,3]	12	3.789	−13.101	150	85.37	33.69	−5.820	151.56
[4,4,4,4]	340	197.36	−0.335	72.16	94.13	7.42	−2.173	72.28
[25,5,5,5]	3900	13540.67	−0.7111	6.92	106.25	25.35	−15.926	27.05
[25,5,4,3]	2150	8039.81	−1.013	7.14	92.12	4.06	−0.713	8.32

Table 6.3: Sensitivity analysis of branching factors in terms of: branching factors vectors, number of nodes of the first optimization problem's scenario tree, computational time, hedging error (mean and std), cost of hedging (mean and std), profit and loss (mean and std).

P&L with mean −4.281 and standard deviation 21.56 is considered. This distribution is valid for every branching factors configuration, since the test scenarios are the same and, consequently, the unhedged performance does not vary. In particular, to guarantee that all configurations of branching factors vectors are tested on the same real-world simulated scenarios, market values were simulated using a time step related to the least common multiple of the time steps of all branching factors vectors: in the analyzed case, according to Table 6.3, l.c.m(3,2,4,5) = 60, so $dt = \frac{T}{60}$. Each configuration then uses the values corresponding to its actual time step, e.g. [10,5,4] uses $dt = \frac{T}{3}$, and so on.
The number of nodes reported in the table represents the size of the first tree built during the stochastic optimization process, i.e. the one used to make the portfolio

construction decision at time $t = 0$.
Starting from the number of nodes and time resources, it is evident that increasing the branching factor generally leads to a higher number of nodes and longer computational times. For example, the configuration with the highest number of nodes (3900) is [25,5,5,5] and it is also the one with the longest computational time (13540.67 s): this highlights the growth in complexity with deeper trees. Meanwhile, the simplest structure, i.e. [3,3], has only 12 nodes and requires only 3.789 s to complete the overall hedging problem.
Regarding the hedging error, i.e. the difference between the hedging portfolio value and the derivative payoff at maturity, smaller values indicate more accurate hedging strategies. The configuration [25,15] achieves the lowest mean error (0.071) with a relatively moderate computational time (74.42 s), suggesting an efficient trade-off between accuracy and resource use. On the other hand, the [3,3] configuration shows that too simplistic models fail to provide reliable outcomes.
Since the hedging problem chosen for the current analysis is self-financing, the unique hedging cost should be the one related to the first construction of the hedging portfolio, that is, at time $t = 0$. To verify the correctness and consistency of the strategy obtained from stochastic optimization, the initial wealth was not constrained to the theoretical Monte Carlo price of the target asset. Instead, the solver was allowed to choose the quantity it found most suitable. Clearly, if the price returned by the solver (reported in Table 6.3 as Cost) is similar to the theoretical one, the hedging strategy can be considered consistent and effective. Given that the Monte Carlo price of the target put option is \$92.367, almost all configurations of branching factors show satisfactory results, with few exceptions. The [3,3] case is particularly notable, as the small scenario tree size leads to an underestimation of the hedging portfolio cost. On the contrary, the [25,5,5,5] configuration overestimates the price obtained from the Monte Carlo simulation. This is also reflected in the profit and loss, which is worse compared to the other cases.
By comparing the performance of [25,15] with that of [3,3], it can be concluded that the issue does not lie in the low number of rebalancing dates (which, in this case, are just two — the initial time and the first step), but rather in the limited number of tree nodes, which should capture a wider range of market conditions: with only 12 nodes (and just 9 scenarios at maturity) the stochastic solver does not have enough information to make reliable predictions about future market movements.
A further interesting comparison concerns the [10,5,4] and [4,4,4,4] configurations. Although the latter has a slightly larger number of nodes and one additional time step, the former achieves better accuracy. This difference arises from the way trees are constructed beyond the initial time step: starting from an initial branching factors vector [10,5,4], the second time step employs a new tree with configuration [10,5] and at the final rebalancing point each node will have exactly 10 children.

This backward erosion of branching factors ensures that, just before maturity (i.e. the final rebalancing opportunity) there are enough scenarios for an effective rebalancing decision. Although this mechanism of erosion is also present in the [4,4,4,4] configuration, maintaining a constant and low branching factor results in an insufficiently reliable simulation, which will lead to suboptimal performance.
It can also be observed that the [25,5,5,5] configuration has a computational demand which is not justified by the improvement in results. A slightly less complex configuration as [25,5,4,3] achieves similarly high-quality outcomes at a lower computational cost.

An alternative method for handling branching factors involves ensuring that all decision stages have the same number of scenarios available in their scenario tree. A total of 1000 Monte Carlo replications are performed for the [25,3,3] and [10,5,3] configurations to analyze this alternative method, whose results are benchmarked against the previously adopted approach, as reported in Table 6.4. An initial wealth

	1st step scenarios	**2nd step scenarios**	**3rd step scenarios**	**time**	**P&L**	
					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 6.4: Comparative overview between the classical method (blue) and the one with constant number of scenario trees (pink).

equal to the target asset's price is fixed in the optimization hedging problem: thus, hedging error and profit and loss coincide.
Starting from [10,5,3] at $t = 0$, the second scenario tree (i.e. the one constructed at $t = 1$) will have [30,5] as branching factors vector, while the last step will handle a tree with a [150] configuration. By doing so, each tree admits the same number of scenarios, unlike the classical method where, as time progresses, the number of scenarios decreases. Surprisingly, this new method does not lead to great improvements in performance, but it results in a significant increased computational time: this is justified by the fact that each scenario tree in subsequent steps is larger than in the classical method. Therefore, while this method ensures a consistent number

of scenarios across all decisional steps, its lack of notable performance improvements and the increase in computational time may limit its practical application. The Python implementation code is presented below.

```
# For a constant number of generated scenarios in subsequent trees,
    use this bf:
if current_time == 0:
    bf = branching_factors
else:
    bf = branching_factors[:-current_time]
    bf[0] *= np.prod(branching_factors[-current_time:])
```

6.3 Financing analysis

The focus now is shifted to the different types of optimization problems that can be solved at each rebalancing step, in relation to what discussed in Chapter 2. Thus, the purpose of this section consists in illustrating the performance of self-financing and non-self-financing cases, in combination with the possibility or not of withdrawing money from the strategy. All the four possible financing combinations are tested over the same 1000 Monte Carlo replications for the case of an Asian put option with the same characteristics described in subsection 6.1.2. At the first stage, the initial wealth is fixed, following constraint (2.4). For subsequent optimization problems, the cash balance for node n_0 assumes the different forms described in Chapter 2.

The considered branching factors vector is [6,4,3,2], which means that four rebalancing times are available until maturity. In this particular case, 144 total scenarios are maintained constant for each scenario tree generated during the optimization task.[1]

Section 6.1 has already overviewed the self-financing case without the possibility of withdrawing money.

Non-self-financing, without withdrawal possibility

This variant of the optimization problem admits an initial wealth to construct the portfolio, but moving forward in time, it is possible that the money obtained from selling assets results to be insufficient to meet all liabilities. Therefore, an additional capital contribution is required, which is considered a rebalancing cost. The empirical distribution obtained is depicted in Figure 6.8. To have a look

[1]See section 6.2 for details on the constant number of scenarios.

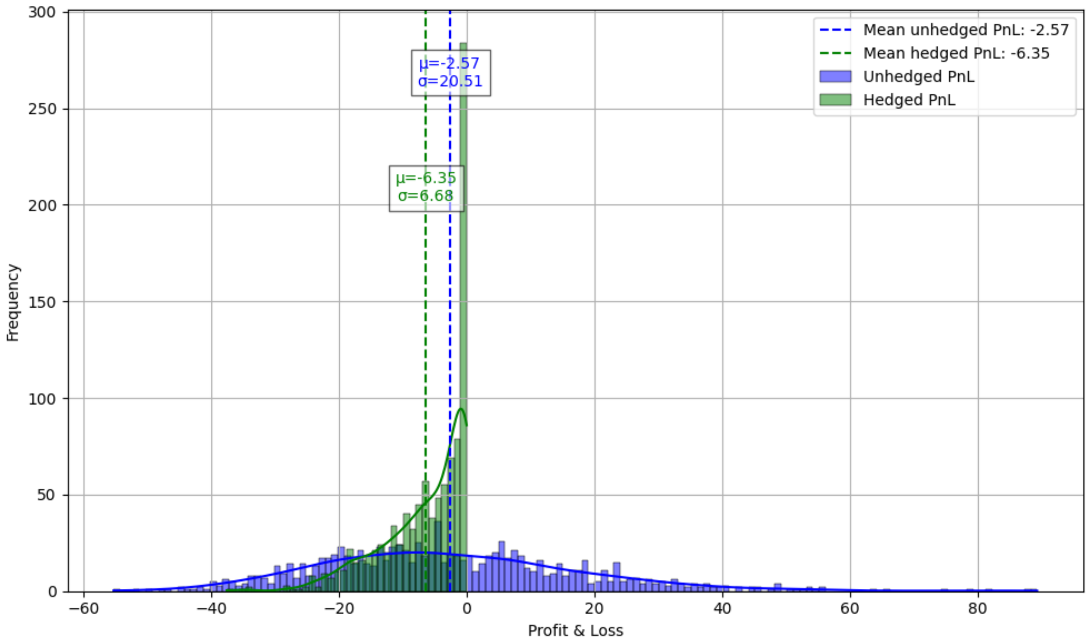


Figure 6.8: Distribution of P&L for the non-self-financing hedging problem without withdrawal possibility.

on what impact has the non-self-financing strategy, Table 6.5 summarizes some important statistical properties of rebalancing costs at intermediate time steps. For each rebalancing date, mean and standard deviation of additional cash flows

	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 6.5: Statistics of intermediate rebalancing costs for the non-self-financing hedging problem without withdrawal possibility.

are computed, as well as their frequency, which is in fact the ratio between the number of Monte Carlo simulations where funds are required in that specific date to the total number of replications. What is most interesting is the total additional cost that the hedger has to sustain if implementing this strategy. The total cost's statistics, derived from the whole hedging horizon, are also provided in Table 6.5. In terms of pure replication, results are satisfactory: with a mean of -0.001 and a standard deviation of 0.0323, the hedging error reflects the great effectiveness of this hedging strategy. However, rebalancing costs have an impact on the P&L distribution, which presents a larger variability (a standard deviation of 6.68) compared to the self-financing case. Thus, it can be concluded that through a non-self-financing strategy is guaranteed that the hedger will not incur significant losses, but this introduces increased risk in terms of final P&L. To be clear, a rebalancing cost equal to zero does not imply that no rebalancing has occurred: rebalancing may have taken place without requiring additional capital.

Self-financing, with withdrawal possibility

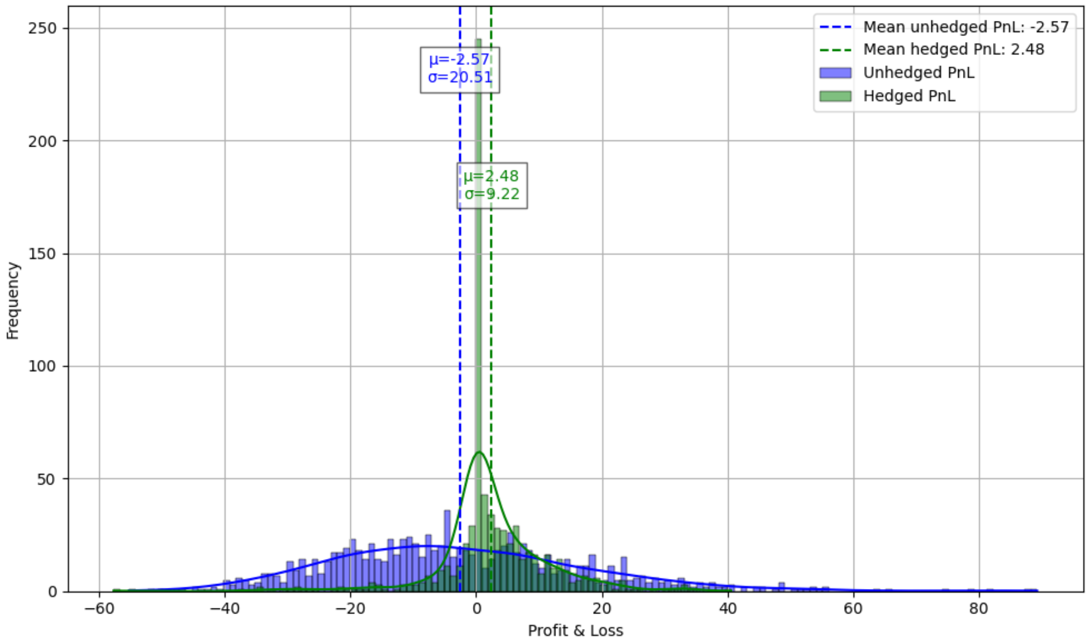


Figure 6.9: Distribution of P&L for the self-financing hedging problem with withdrawal possibility.

This variant of the hedging problem admits that potential surplus money can be withdrawn by the hedger after rebalancing, resulting in gains along the hedging horizon that may be reinvested in other activities.[2].

Figure 6.9 shows the behaviour of the P&L distribution, which is not the best so

[2] See section 2.3 for details.

far: on average, this strategy provides a gain (mean 2.48), but at the cost of a higher variability (std 9.22).
Additionally, with a mean of -2.046 and a standard deviation of 6.382, the hedging error is still low but exhibits a certain level of variability, implying a residual risk of not fully covering the target option's payoff. This makes this configuration less accurate than the previous ones.
Meanwhile, Table 6.6 provides an insight into the statistical properties of the withdrawal operations during the hedging horizon. As in the previous case, attention should be paid to the total gain's characteristics: on average, there will be an additional gain of \$4.495 at the end of the time horizon due to the possibility of withdrawing excess cash from the strategy.

	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 6.6: Statistics of intermediate gains for the self-financing hedging problem with withdrawal possibility.

Non-self-financing, with withdrawal possibility

The last configuration to be analyzed allows for additional cash flows, both inflows and outflows, at the rebalancing steps (excluding the initial time). Table 6.7 presents the most relevant properties of the intermediate steps' additional cash flows. Due to the fact that all kinds of cash flows are allowed, both negative and

	decision stages			
	stage 1	stage 2	stage 3	total
mean cash flow	-3.238	-3.158	-0.769	6.993
std of cash flows	9.586	7.536	4.576	12.641
cash flows frequency	100%	100%	100%	

Table 6.7: Statistics of cash flows at intermediate stages for the non-self-financing hedging problem with withdrawal possibility.

positive, this strategy leads to the worst empirical distribution for the P&L, which presents mean −7.95 and standard deviation 14.40. By contrast, the hedging error shows outstanding results, confirming that the strategy can hedge the short position in the target asset, with mean 0.002 and standard deviation 0.248.

Examining the rebalancing frequencies of the discussed strategies, it is clear that additional cash flows are more common in the initial steps and gradually decrease as maturity approaches. Similarly, the involved cash flow amounts are typically higher at the beginning of the hedging horizon and decline over time. Such behaviour is encouraging: while the strategy may require some adjustments just after the initial time, subsequently it is able to track the target asset sufficiently well using only portfolio's internal funds.
To provide a comprehensive overview of both the hedging error and the profit and loss, Table 6.8 is presented. In conclusion, while allowing for additional intermediate cash flows may offer advantages in terms of replication, it may not yield significant benefits from a P&L perspective. In fact, the strategy without any external cash flows during the intermediate steps, i.e. the self-financing strategy without withdrawal possibility, achieves the best P&L outcome (which coincides with the hedging error) and ensures a good hedging at maturity. On contrast, the best hedging error distribution, provided by the non-self-financing without withdrawal possibility variant, is actually associated with a higher profit and loss. The key objective remains minimizing the P&L variance with respect to the unhedged case: incurring a minor loss is acceptable if it leads to lower risk by a reduced variance.

self-financing	**withdraw**	**mean he**	**std he**	**mean P&L**	**std P&L**
✓	✘	1.37	5.30	1.37	5.30
✘	✘	−0.001	0.03	−6.35	6.68
✓	✓	−2.05	6.38	2.48	9.22
✘	✓	0.002	0.248	−7.95	14.40

Table 6.8: Statistical overview of hedging error (he) and profit and loss (P&L) for the considered financing strategies. The highlighted values represent the best distributions.

6.4 Risk aversion

Recalling what was discussed in section 2.1, attention is now shifted to the risk aversion parameter γ, which enables the adjustment of the weight assigned to the positive hedging error, thus generating an asymmetric objective function. The maximum penalty allowed is 1 and remains fixed for the negative error, reflecting the priority of avoiding losses. Meanwhile, the gamma parameter is varied within the set $\{0, 10^{-13}, 10^{-12}, 10^{-11}, ..., 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. The aim is to show how hedging results can vary when negative and positive deviations are penalized differently. The evaluation begins at the extreme case where γ is set to zero, resulting in penalizing only the shortfall, and extends to $\gamma = 1$, which leads to the symmetric case. The simulation setting consists of 1000 Monte Carlo replications to perform a self-financing hedging of an Asian put option, with no opportunity to withdraw money and a fixed initial wealth equal to the target asset premium. For each γ, P&L empirical distribution's mean and standard deviation over the 1000 replications are computed and illustrated in Figure 6.10.

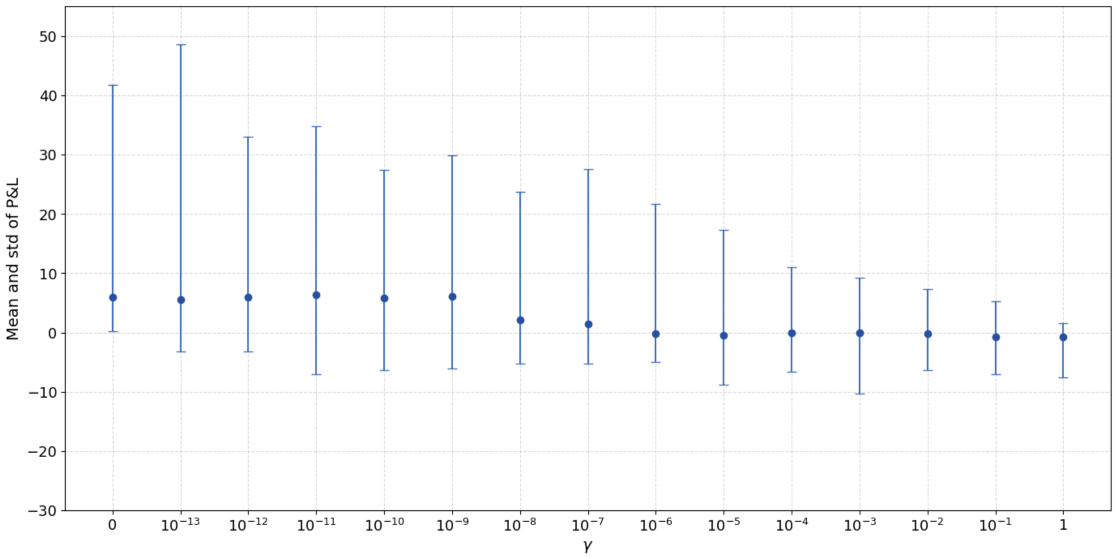


Figure 6.10: Comparison between P&L distributions for different risk aversion parameter γ, based on 1000 replications.

An additional information that can be inferred from the plot is the skewness of the empirical distributions, which becomes evident through the distribution's asymmetry with respect to the expected value.
As expected, for decreasing values of γ the profit and loss distribution tends to shift towards positive values; in contrast to the symmetric case, assigning a larger penalization coefficient to the shortfall allows for the possibility of positive

deviations, which, as gamma decreases, are less penalized. This results in an overall gain in terms of P&L; thus, the distribution is shifted towards positive values rather than being centered around zero. Clearly, this also impacts variability, leading to a higher standard deviation. However, the increased variance is associated with positive values, which implies that the risk of losses is still reduced.
The obtained statistical properties for some analyzed γ are listed in Table 6.9: they numerically show that, for small values of surplus penalization, profit and losses present asymmetrical distributions, more centered and shifted towards positive values. The skewness values are particularly interesting, since they confirm the previous observations: increasing positive skewness values point out that P&L distributions exhibit heavier upper tails.

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 6.9: Statistical properties (mean, standard deviation and skewness) of P&L empirical distribution for different values of the risk aversion parameter γ.

An interesting aspect to consider when giving less importance to the surplus is the initial cost required by the hedging strategy to construct the portfolio. In the absence of a fixed initial wealth, the strategy may choose a surprisingly high capital in order to be able to better mitigate negative errors at maturity. This is justified by the fact that, as γ decreases, the objective at maturity progressively becomes the avoidance of only losses, thus allowing for the possibility of net profit. This aspect is illustrated in Figure 6.11, generated under the same setting conditions as before, but with the optimizer free to select the portfolio's initial cost. In this plot, the empirical distribution of the initial capital required by the strategy is compared with two benchmark prices: the theoretical initial price of the target asset and the price of the super-replication strategy.[3] As shown in Figure 6.11 and Table 6.10, when the objective function is symmetric (i.e. when $\gamma = 1$) the initial investment required by the hedging strategy is comparable to the initial price of the target asset. As γ decreases, the initial wealth increases, converging towards the price provided by the super-replication strategy, which represents the theoretical upper bound of the cost for constructing the hedging portfolio.

[3] See section 2.4 for theoretical details.

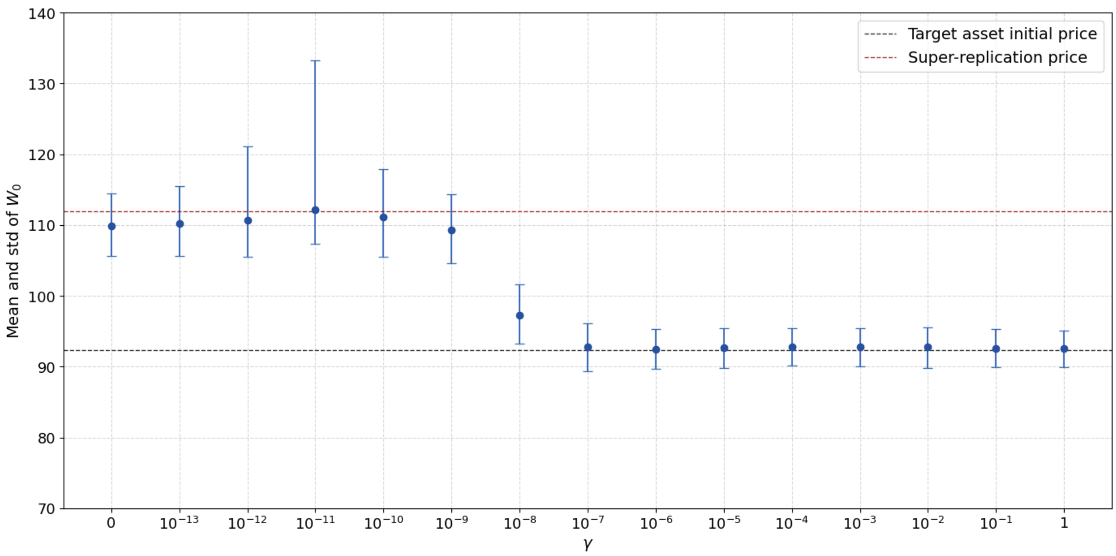


Figure 6.11: Empirical distributions of the initial wealth W_0 required by the hedging strategy, represented by their mean and standard deviation, for different values of the risk-aversion parameter γ.

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 6.10: Statistical properties (mean, standard deviation and skewness) of W_0 empirical distribution for different values of the risk aversion parameter γ.

6.5 Pricing through hedging

It can be interesting to analyze what would be the initial price required by the hedging strategy if the solver of the hedging optimization problem is allowed to choose the initial wealth W_0 to construct the portfolio at time $t = 0$. To analyze this aspect, a series of Monte Carlo real-world simulations is performed, considering the formulation (2.10). In most of them, the hedging strategy requires an initial capital that is very close to the price of the target asset at the beginning of the hedging horizon. This demonstrates that, in majority of cases, the premium of the derivative to be hedged is sufficient to construct a valid strategy. In contrast, a too-low estimate reflects insufficient hedging, while an overestimated value suggests

that the hedging is more than necessary. In both cases, a potential loss occurs: in the former because the hedge is inadequate and the short position remains exposed; in the latter because the strategy is unnecessarily expensive.
The detailed analysis conducted in this section focuses on the comparison between the initial wealth required by hedging and the theoretical initial price of the target asset, which is computed through

- Black-Scholes-Merton model for European vanilla options;
- 10^4 Monte Carlo replications to apply the risk-neutral valuation principle, averaging the discounted payoff, for Asian options.

Through a self-financing problem, where funds cannot be withdrawn and the initial wealth is not fixed, the W_0 distribution and its comparison with the theoretical price can be studied.
Figure 6.12 shows the results of this analysis made on a European vanilla call option, whose Black-Scholes price at $t = 0$ is \$26.013, illustrating how the hedging strategy prices are distributed over 1000 Monte Carlo test replications.

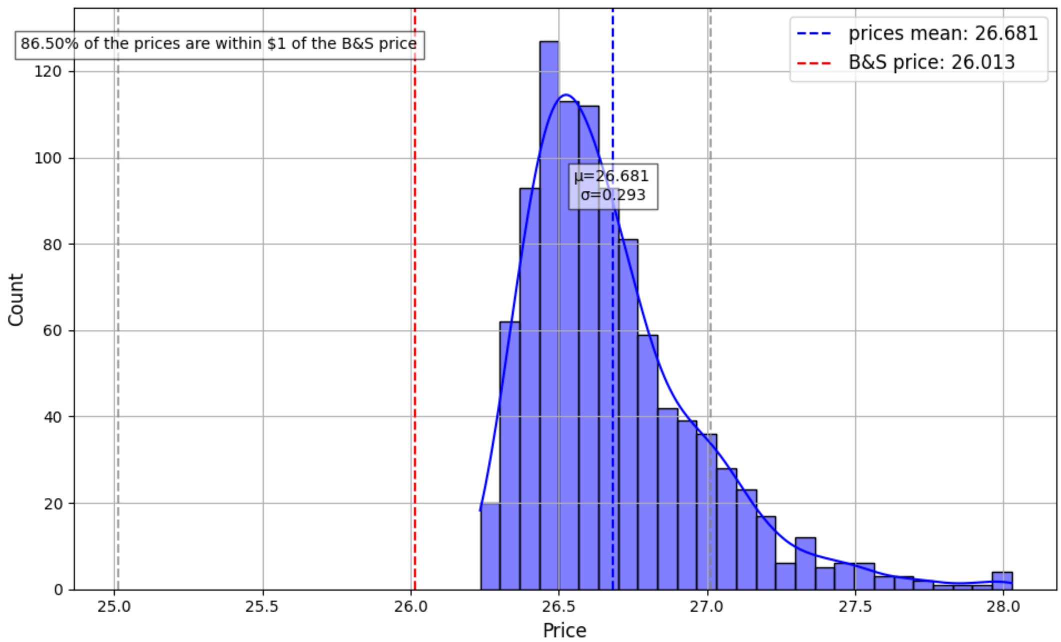


Figure 6.12: Distribution of the hedging strategy's prices required for a European vanilla call option and its comparison with the target asset theoretical price (dotted red lines).

An interval with a radius of \$1 centered around the Black-Scholes price (vertical red line) is marked by gray dotted lines. In this specific case, 86.50% of the prices required to hedge the target asset fall within this interval. The chart confirms that, if the W_0 distribution's mean (26.681) is regarded as an estimator of the theoretical price of the derivative, the implemented optimization problem successfully estimates it in the case of European options: taking into account the standard deviation of 0.29, the mean falls within a very narrow range around the Black-Scholes price. The small gap between the compared prices can be explained by the fact that the Black-Scholes-Merton model assumes a frictionless market, which does not account for transaction costs. In contrast, the implemented stochastic optimization method incorporates them into the model. Their impact proves to be minimal, since they amount to just 1%.

To present the same analysis in the case of an Asian put option, whose Monte Carlo risk-neutral price is $92.362., Figure 6.13 is provided below. In this case as

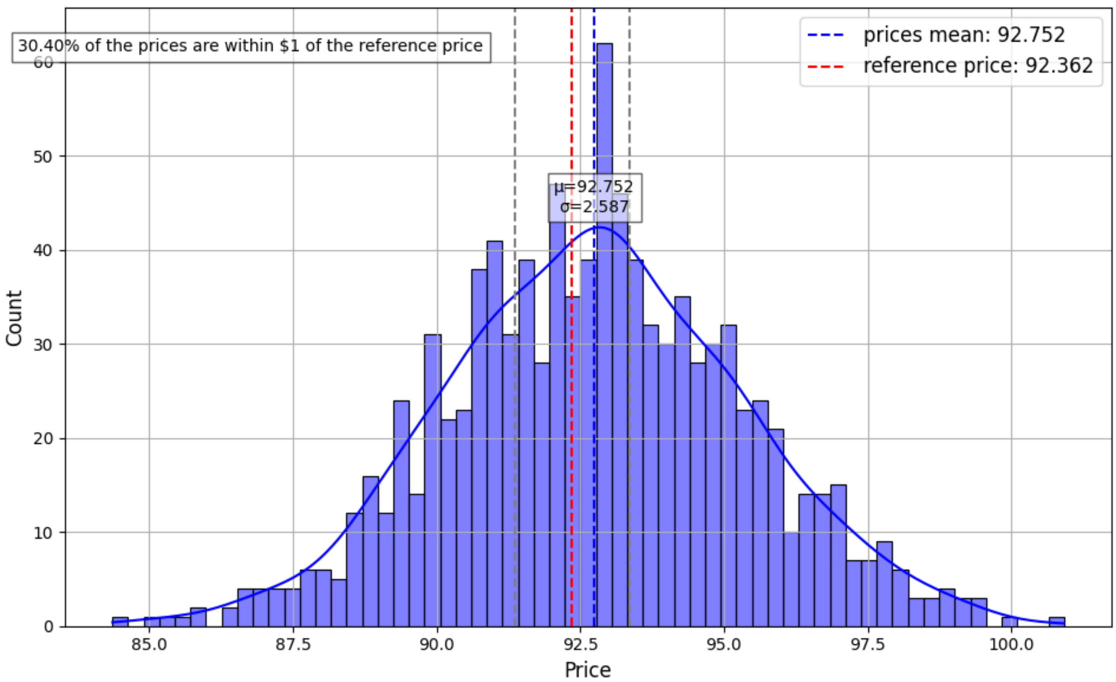


Figure 6.13: Distribution of the hedging strategy's prices required for an Asian put option and its comparison with the target asset theoretical price (dotted red lines).

well, the strategy requires an initial cost similar to the theoretical price in order to effectively hedge the derivative under consideration. As the Asian option is based on multiple underlying assets, this may be reflected in a greater variability in the initial capital required to construct the portfolio, making the estimate less precise:

the standard deviation of the distribution, in this case, is 2.587, which is higher than the vanilla call one, but still acceptable. Nevertheless, in 30% of the cases, the strategy's cost is only \$1 apart from the theoretical price, and in the other 70% the difference is still within acceptable limits.

Table 6.11 provides a comprehensive overview of the analysis conducted in this section.

Target asset	Premium	Price estimator		Prices within \$1
		μ	σ	
Vanilla call	26.01	26.68	0.29	86.50%
Asian put	92.362	92.752	2.587	30.40%

Table 6.11: Key values for the pricing through hedging analysis: the estimate obtained from stochastic optimization is compared to the theoretical premium of the target asset.

6.6 Transaction costs analysis

Since trading financial instruments entails transaction costs, the stochastic optimization problems outlined in Chapter 2 include the costs c_j for each hedging asset $j \in \mathcal{A}$ to better capture real-world conditions. Transaction costs are assumed to be proportional to the amount of money involved, i.e. if a quantity $x_{j,t}$ of assets of type j is traded at time t with price $p_{j,t}$, the cost related to the transaction is equal to $c_j x_{j,t} p_{j,t}$, which yields to a total cost of

- $(1 - c_j)x_{j,t}p_{j,t}$ if assets j are purchased;
- $(1 + c_j)x_{j,t}p_{j,t}$ if assets j are sold.

This section investigates how the optimizer responds to changes in transaction costs, starting from zero and reaching up to 10%. The chosen optimization setting is (2.1) − (2.9): self-financing, with fixed initial wealth and no withdrawals allowed. An Asian put option is used as target asset, with strike $K = 250$ and maturity $T = 1$ year. The selected transaction costs, assumed to be the same for all hedging assets (except for the bank account, which has no cost), are $\{0, 10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}\}$.
The resulting P&L's means and standard deviations are depicted in Figure 6.14. The plot shows a stable behaviour and highlights the robustness of the stochastic

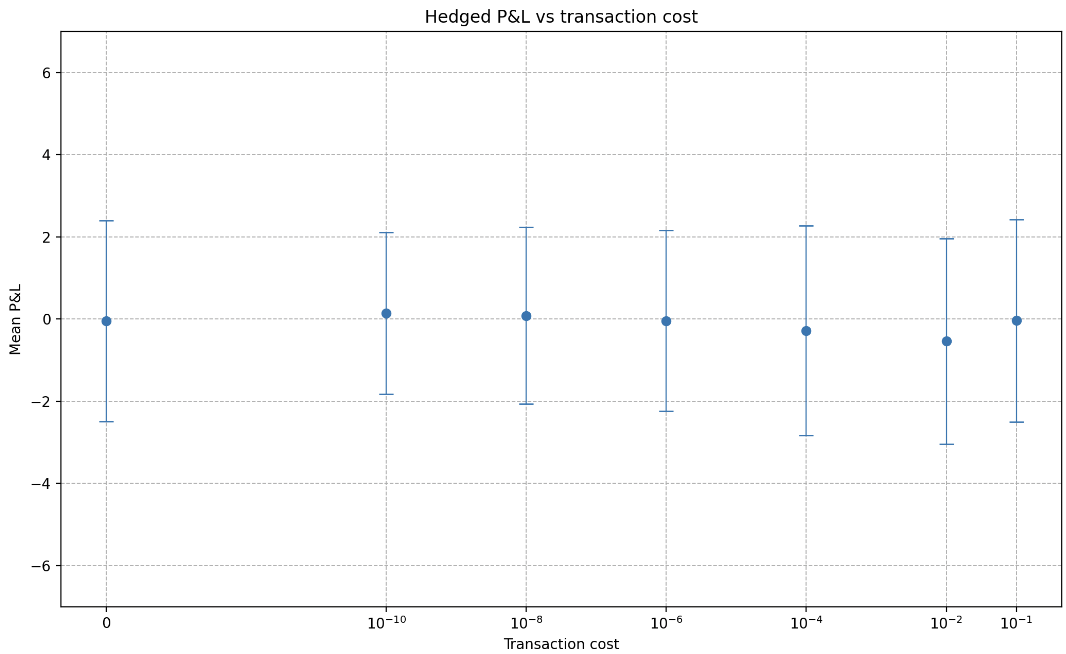


Figure 6.14: Impact of transaction costs on profit and loss, represented by mean and standard deviation.

optimization approach in the presence of transaction costs as well as in their absence. In particular, it demonstrates that stochastic optimization remains stable under transaction costs: the P&L distribution is centered around zero for all cases; more importantly, it exhibits a low variance, ensuring a successful reduction of risk despite higher costs. Moreover, the increase in transaction costs does not influence the computational time required to resolve the hedging problem. This makes the stochastic approach one of the most effective methods for addressing hedging problems.

6.7 Stochastic models comparison: GBM vs MM

Chapter 3 presented two alternative stochastic models for the simulation of stock paths in scenario trees. A comparative analysis is now conducted to assess their performance in terms of both computational time and hedging effectiveness, considering the three different implementation discussed in section 5.3. To avoid arbitrage opportunities, the absence of dominant strategies approach is chosen for this analysis.
An Asian put option with strike $K = 250$ and $T = 1$ year is selected to asses the solution quality of the same hedging strategy (self-financing, without withdrawal

possibility, with fixed initial wealth) performed with the three previously described stochastic model variants.
As shown in Figure 6.15, Figure 6.16 and Figure 6.17, all three cases result in satisfactory performance, successful hedging of the short position in the target asset and improvement of variability over the unhedged case (with a standard deviation of 11.26). However, some differences can still be observed. Moment Matching stochastic model proves to be the most time-consuming, requiring 4423.73 s to complete the overall hedging problem. The Gurobi-based approach is instead the fastest one, highlighting Gurobi's efficiency as an optimizer. Comparing the two GBM alternatives, although they exhibit similar P&L, the model using Gurobi outperforms the other in terms of computational demand.

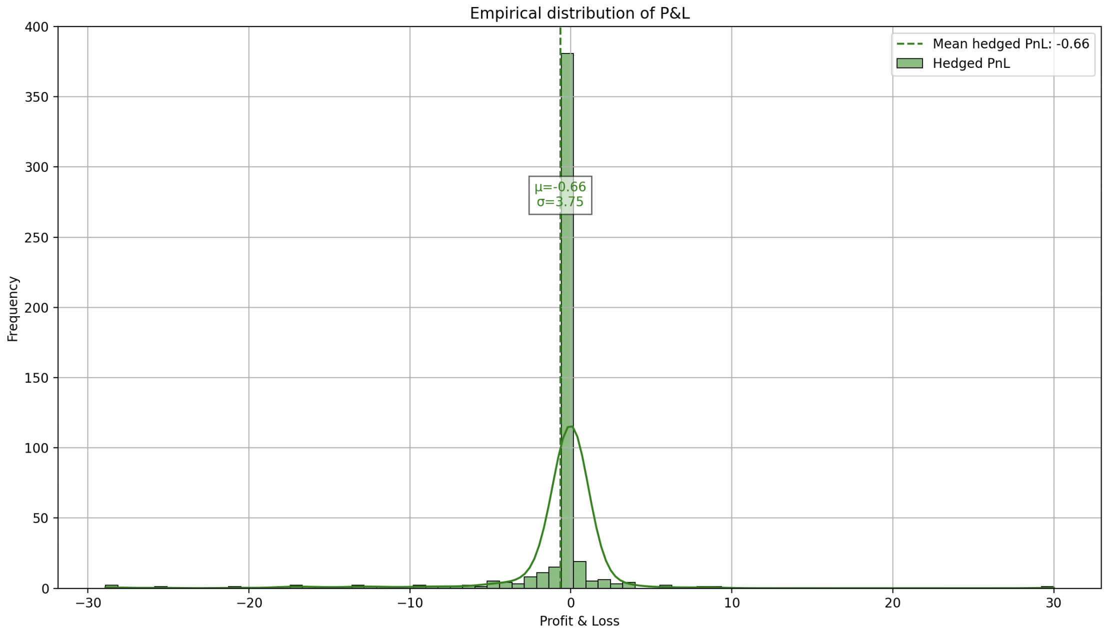


Figure 6.15: P&L with GBM – SLSQP: 746.18 s required; outliers detected and discarded: 1.8%.

Despite the higher computational burden, the Moment Matching method achieves a superior level of accuracy that through the GBM model can be reached, in general, with a more complex branching factors configuration (in this analysis, [15,5,4] is considered). What seems to contribute to the success of MM is its effort to align not only the first and second moments and the correlations, but also the skewnesses, resulting in better overall outcomes.

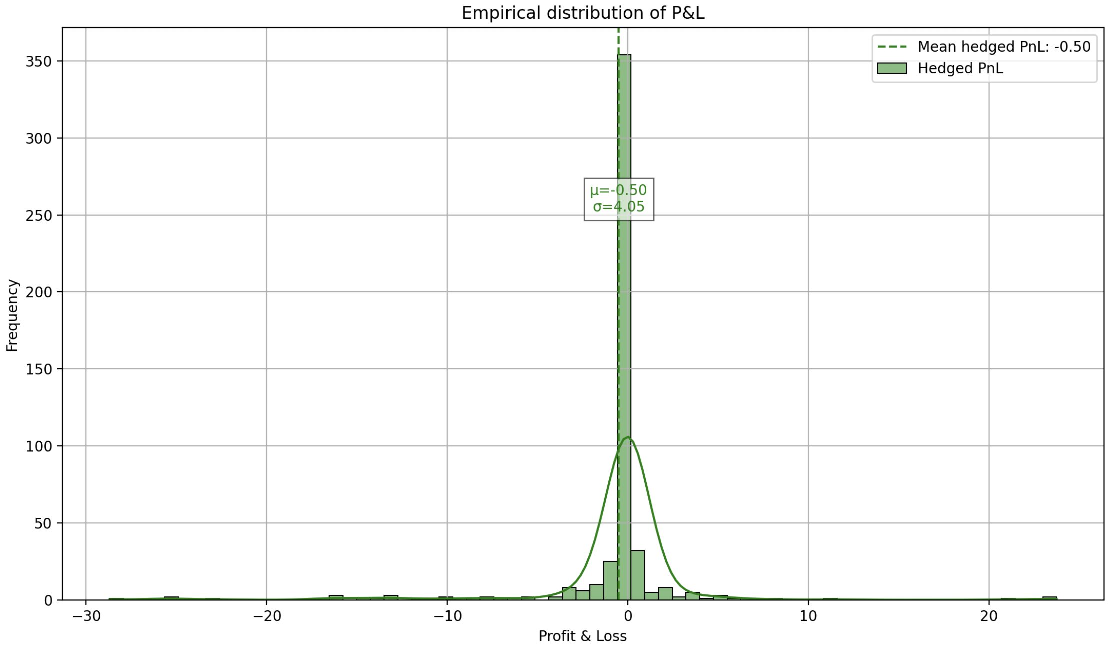


Figure 6.16: P&L with GBM – Gurobi: 277.85 s required; outliers detected and discarded: 1.8%.

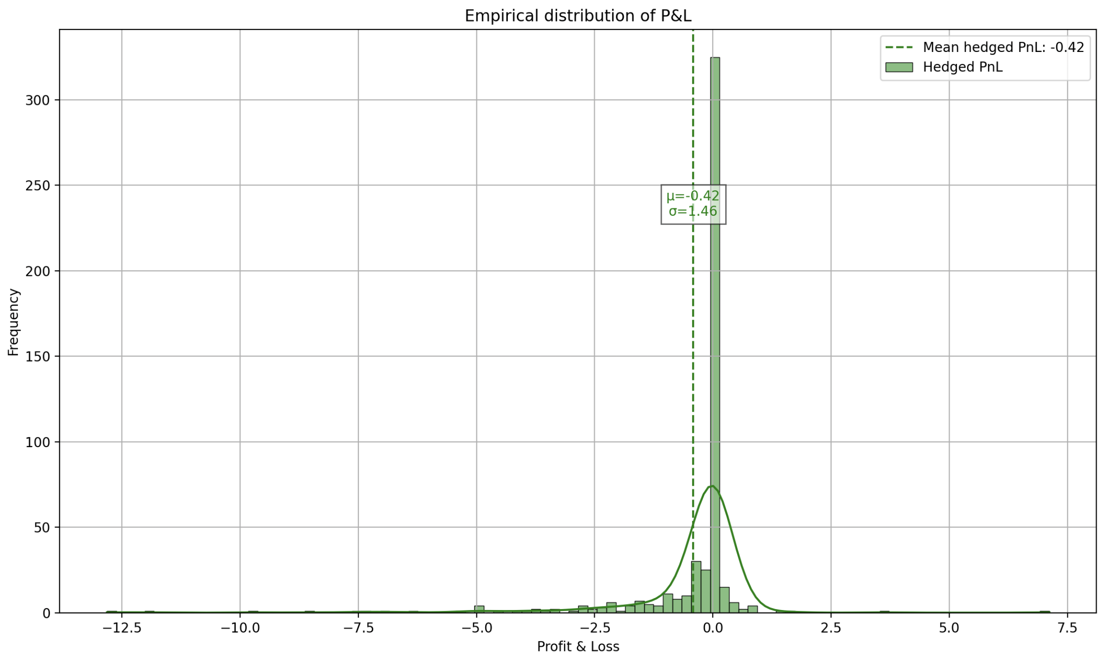


Figure 6.17: P&L with MM: 4423.73 s required; outliers detected and discarded: 0.4%.

In conclusion, while achieving great P&L performance is important, computational efficiency becomes equally essential when the problem complexity increases (due to more time steps or to a less trivial target asset). Thus, GBM stochastic model solved with Gurobi demonstrates to be the best choice: it achieves satisfactory P&L distribution, while requiring minimal computational time.

6.8 Benchmark: Delta hedging

This section will specifically focus on the comparison between the two hedging approaches introduced in Chapter 2. In particular, stochastic optimization and delta hedging will be analyzed in the specific case of a European-style vanilla put option. The benchmarking framework involves a comparison of self-financing strategies, using branching factors typical of the binomial model (e.g. [2,2,2,2,2,2,2,2]). Assuming the Black-Scholes-Merton model, the delta of a European vanilla put option at time t is given by $\Delta_t = N(d_1) - 1$, considering d_1 as defined in (4.4). Under the implemented delta hedging strategy[4], the positions in the hedging assets $(x_{c,t}, x_{s,t})$, respectively for cash and the underlying, are updated at each time t according to the following procedure:

$$x_{s,t} = \begin{cases} \Delta_t & t = 0, ..., T-1 \\ \Delta_{T-1} & t = T \end{cases} \tag{6.3}$$

$$x_{c,t} = \begin{cases} \dfrac{w_0 - x_{s,1} S(1)(1+c)}{B(0)} & t = 0 \\ \dfrac{x_{c,t-1} B(t) + x_{s,t-1} S(t) - x_{s,t} S(t) - |x_{s,t} - x_{s,t-1}| S(t) c}{B(t)} & t = 1, ..., T-1 \\ x_{c,T-1} & t = T \end{cases}, \tag{6.4}$$

where

- the stock and cash values are identified by $S(t)$ and $B(t)$;
- w_0 is a certain initial wealth used to construct the portfolio at time $t = 0$;
- c represents the transaction cost of the unique put option's underlying.

As previously explained, the delta expresses how many shares are required to hedge the option in a delta neutral manner. Positions are adjusted up to the time $T-1$ preceding maturity, after which no further rebalancing takes place and the hedging

[4] From [1].

strategy's effectiveness is evaluated through the P&L.
In this section, the considered derivative to hedge is a European put option with strike $K = 300$, maturity $T = 1$ year and TSLA as underlying. Transaction costs are included in the analysis.

Several rebalancing times.

An initial comparison is made in a framework that permits a larger number of rebalancings, based on a branching factors vector with 9 time steps. Regarding the stochastic optimization trees, each node will have two children nodes. From a computational time perspective, a significant difference can be highlighted when comparing stochastic optimization and delta hedging: the computational time for the former amounts to 2219.05216 seconds, whereas the latter requires only 0.00303 seconds. This difference essentially reflects the complexity of the two methods. Stochastic optimization is a more advanced technique that involves solving multiple optimization problems and relies on a range of different scenarios, which naturally increases the computational cost. In contrast, delta hedging is a more straightforward strategy, as it avoids optimization problems and simply adjusts the positions at each time t, based on the current values of the hedging assets, following the deterministic procedure (6.3) $-$(6.4). Clearly, this results in a minimal computational effort, particularly because scenario trees simulation is not required to maintain a delta-neutral position.
Figure 6.18 and 6.19 show the P&L for both cases taken into account. Outliers were filtered out in the post-processing stage.
Comparing the two strategies, they both have successfully hedged the risky position. However, delta hedging has a lower average performance and a smaller standard deviation compared to stochastic optimization. This suggests that delta hedging provides a more stable outcome, with more limited changes around the mean: more controlled outcomes are ensured since delta hedging follows a deterministic rebalancing rule. On the other hand, stochastic optimization exhibits greater variability, due to the fact that its objective consists in minimizing the mean: this results in a larger standard deviation, meaning that this approach is able to adapt to different market conditions but at the cost of increased risk.
Shifting the focus to the adaptability to various market conditions, stochastic optimization clearly outperforms delta hedging, especially in complex market conditions. Thus, the choice between the two methods depends not only on computational resources but also on the level of precision and adaptability expected from the hedging strategy. Furthermore, as detailed in Chapter 2, delta hedging requires a specific pricing model for the derivative to be hedged, while stochastic optimization does not rely on this, further confirming its adaptability.

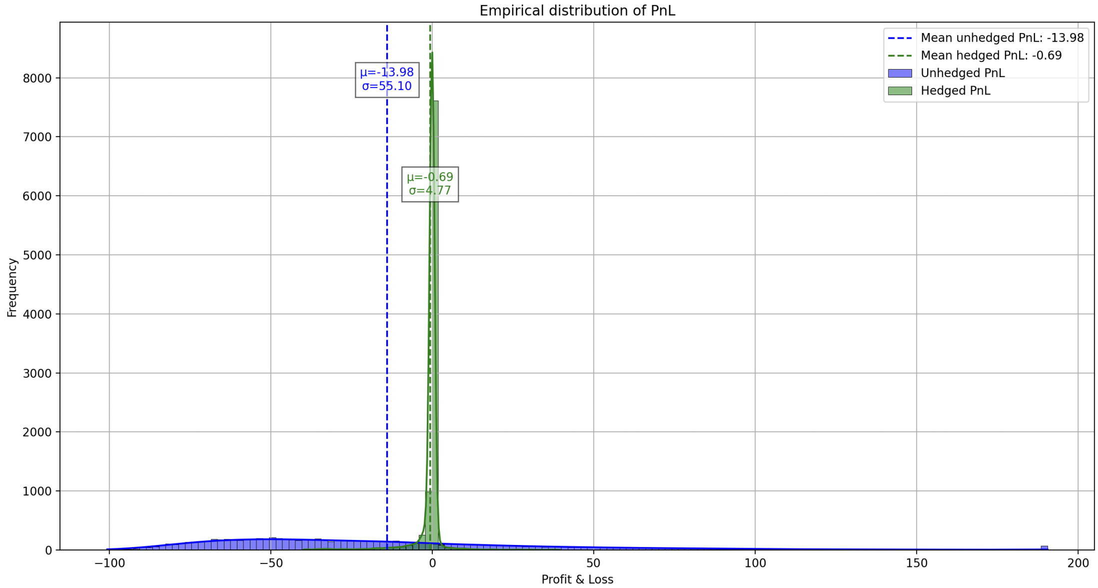


Figure 6.18: Profit and loss resulting from stochastic optimization for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 9 time steps.

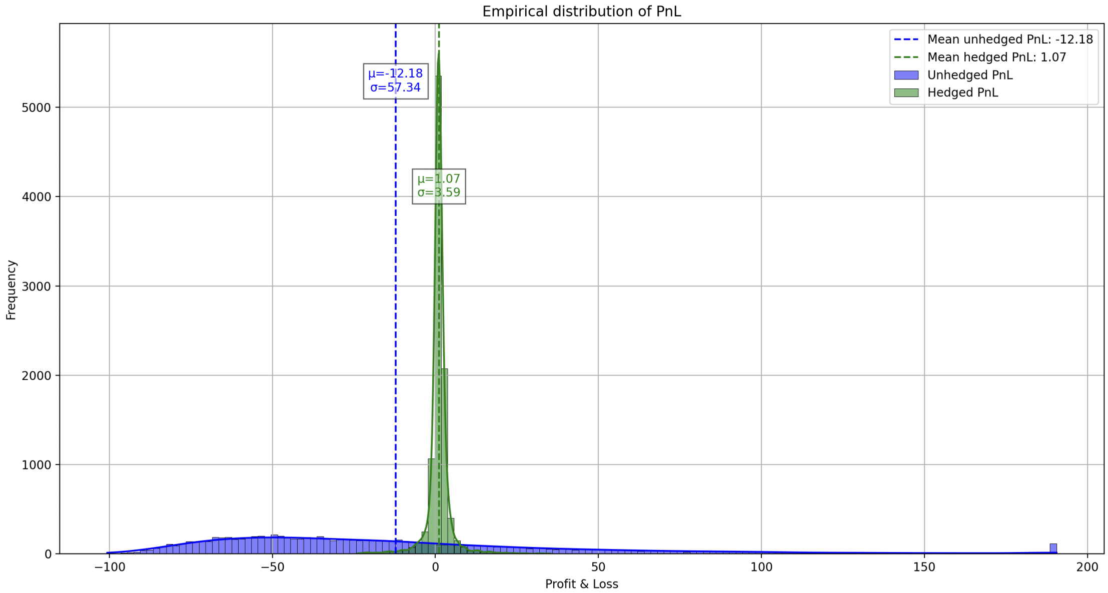


Figure 6.19: Profit and loss resulting from delta hedging for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 9 time steps.

Less rebalancing times.

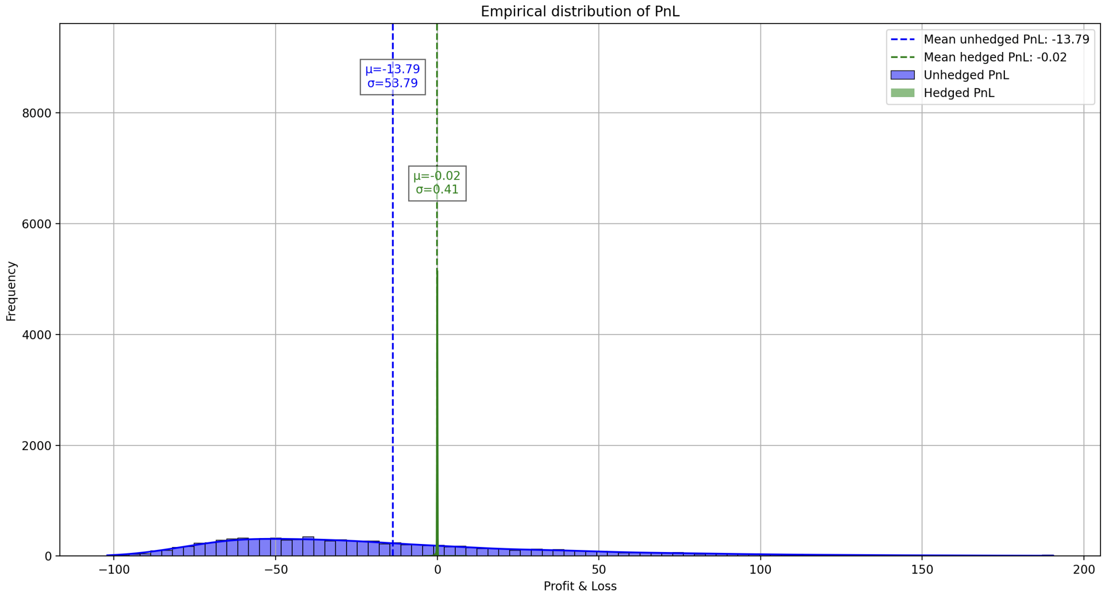


Figure 6.20: Profit and loss resulting from stochastic optimization for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 4 time steps.

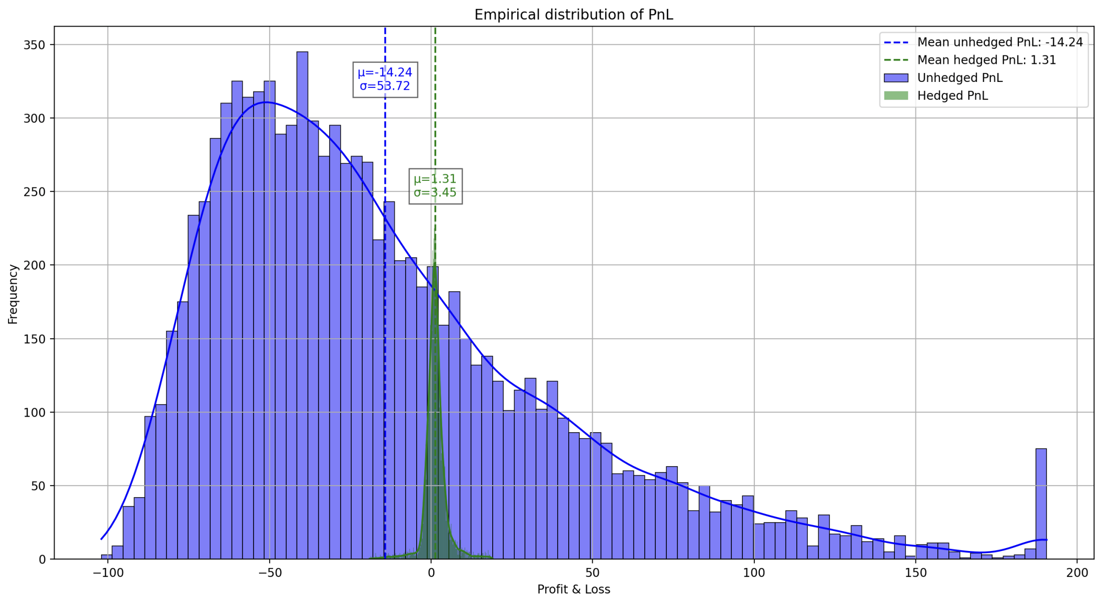


Figure 6.21: Profit and loss resulting from delta hedging for a European vanilla put option with strike $K = 300$ and maturity $T = 1$ year, with 4 time steps.

Figure 6.20 and 6.21 compare the empirical P&L distributions of the two hedging methods under the constraint of having only three rebalancing times, e.g. a branching factors vector [25,3,3]. It is evident that, even with just few time steps, the stochastic optimization approach provides an optimal outcome in terms of distribution, having both the mean and standard deviation very close to zero. On the other hand, delta hedging fails to fully mitigate price fluctuations between rebalancing dates. It is essential to keep in mind that delta hedging is a first-order approximation, which effectively hedges only against small price movements in the underlying asset. However, with a 12-month horizon and quarterly rebalancing, significant stock price variations may compromise delta-hedging performance.
In conclusion, while delta hedging remains a viable strategy, the stochastic optimization approach proves to be a more robust method.

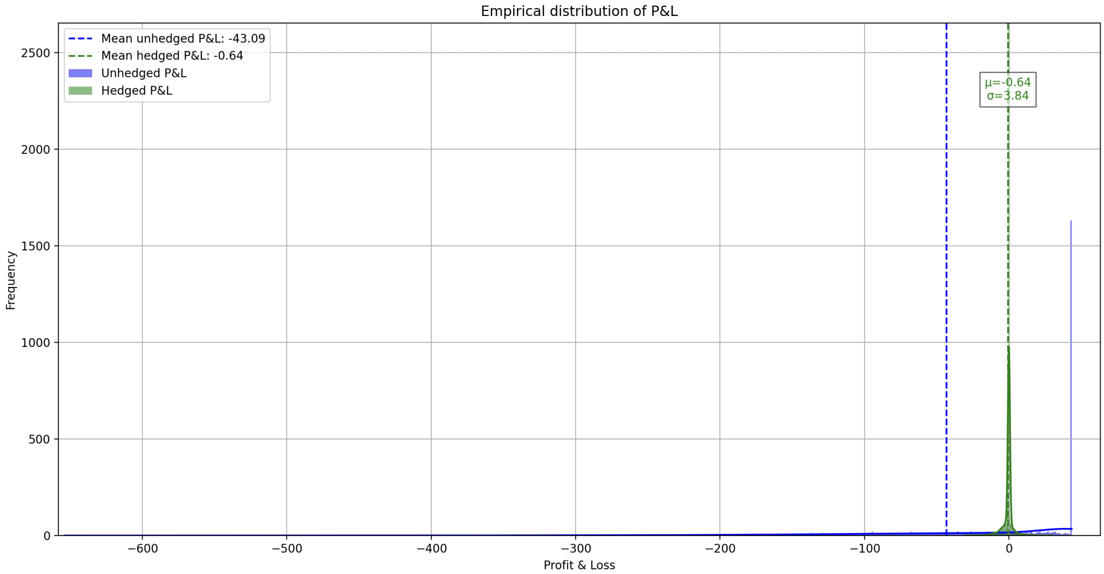
6.9 Barrier Options

A selection of barrier options has been analyzed to highlight interesting behaviours and distinctive features in their valuation and risk profiles. A self-financing, without withdrawal possibility and with initial fixed cost strategy is implemented and tested over 5000 Monte Carlo replications.

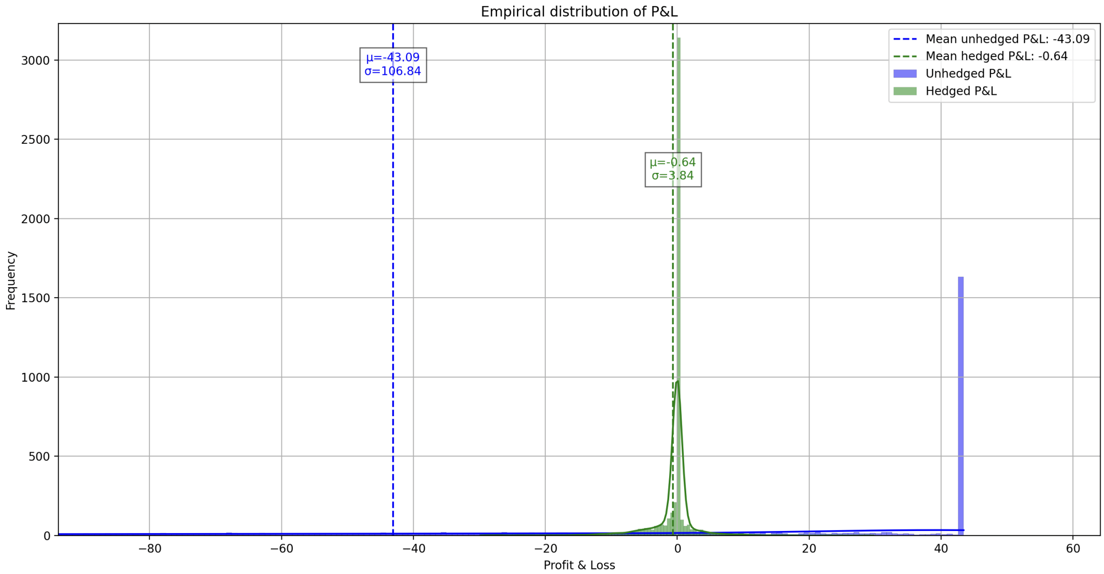
Up-and-in call: P&L and pricing through hedging

Firstly, the focus is on illustrating the capability of stochastic optimization to deal with the hedging of these relatively intricate and non-trivial exotic options. An up-and-in call written on AMZN is chosen, with strike $K = 240$ and barrier $H = 250$. Other two stocks are considered as hedging assets, for a total of three stocks and nine vanilla options (one call and one put on each stock); also a bank account is used as hedging instrument. All their details are listed in Table 6.12.
Also in this case, the adoption of the stochastic optimization strategy yields impressive outcomes in terms of P&L. Figure 6.22 clearly illustrates that in the case of a short position in a call there is no limit to the potential losses that an unhedged position could face: if the underlying asset price increases unexpectedly, the option holder will exercise the option, resulting in great losses for the bank. In contrast, a hedged position not only ensures a smaller average loss (-0.64 versus -43.09), but it also significantly reduces the variability of the potential outcomes.
In the unhedged case, the most frequent outcome is approximately \$43.4, which corresponds to the premium the bank received at $t = 0$ for selling the target asset. This value slightly grown at the risk-free rate since it is held until maturity. This gain can occur in the following cases:

1. the underlying asset price never exceeded the barrier level, meaning the option



(a) Overall P&L distributions, showing the complete range of outcomes.



(b) Zoomed-in section of P&L curves to better show the hedged distribution's shape.

Figure 6.22: Empirical distribution of P&L for an up-and-in call option: unhedged (blue) vs hedged (green) positions.

was never activated, resulting in no loss for the short position;

2. the barrier was hit during the option's lifetime, but at maturity the underlying

	stocks			**strike**	**barrier**
	AMZN	META	GOOGL	240	250
initial price	242.92	102.05	107.19		
put strike	250	110	120		
call strike	230	100	100		

Table 6.12: Up-and-in call features: underlying (the highlighted one), risky hedging instruments, strike, barrier.

price caused the active option to not be exercised, leading to a zero payoff.

It is evident that the selected hedging strategy removes the opportunity to realize this profit. However, it is important to look at the bigger picture: although it may limit net gains in certain scenarios, it definitely reduce the risk of substantial losses. To analyze the initial costs required by the strategy to hedge this type of barrier option, the results of 2000 Monte Carlo replications solving a self-financing problem with an unfixed initial wealth are shown Figure 6.23.

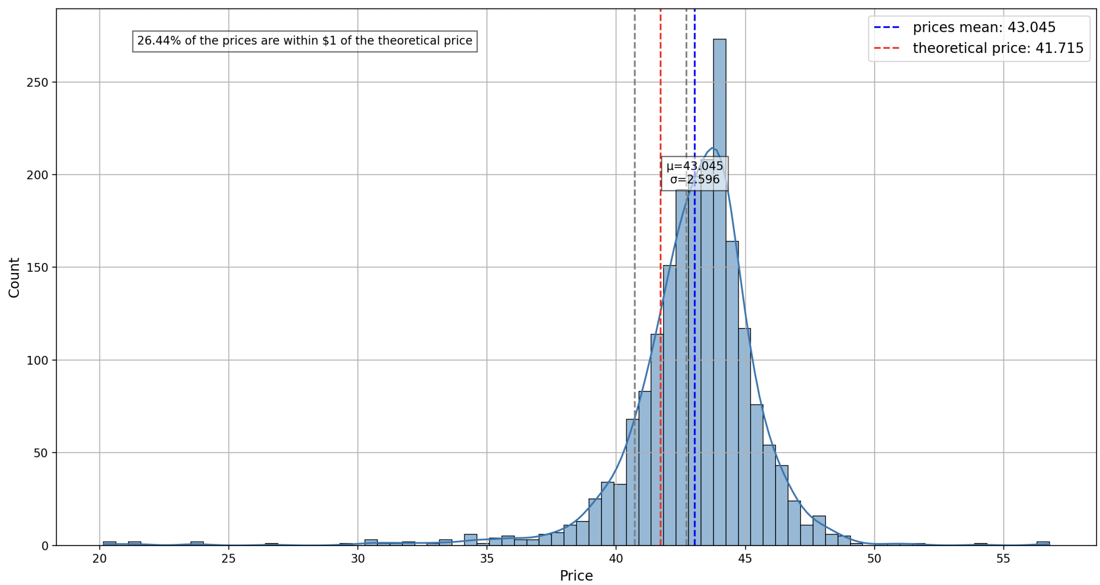


Figure 6.23: Initial wealth distribution W_0 chosen by the optimization solver, compared to the theoretical Monte Carlo price of the considered up-and-in call option.

Also in this non-trivial case, setting up an efficient hedging portfolio requires an initial investment that is very similar to the initial price of the risky derivative to be hedged.

Down-and-in call: strategy's behaviour in different maturity situations

A second type of barrier option, a down-and-in call, is now implemented, with characteristics shown in Table 6.13.

	stocks			**strike**	**barrier**
	AMZN	META	GOOGL	200	240
initial price	242.92	102.05	107.19		
put strike	250	110	120		
call strike	230	100	100		

Table 6.13: Down-and-in call features: underlying (the highlighted one), risky hedging instruments, strike, barrier.

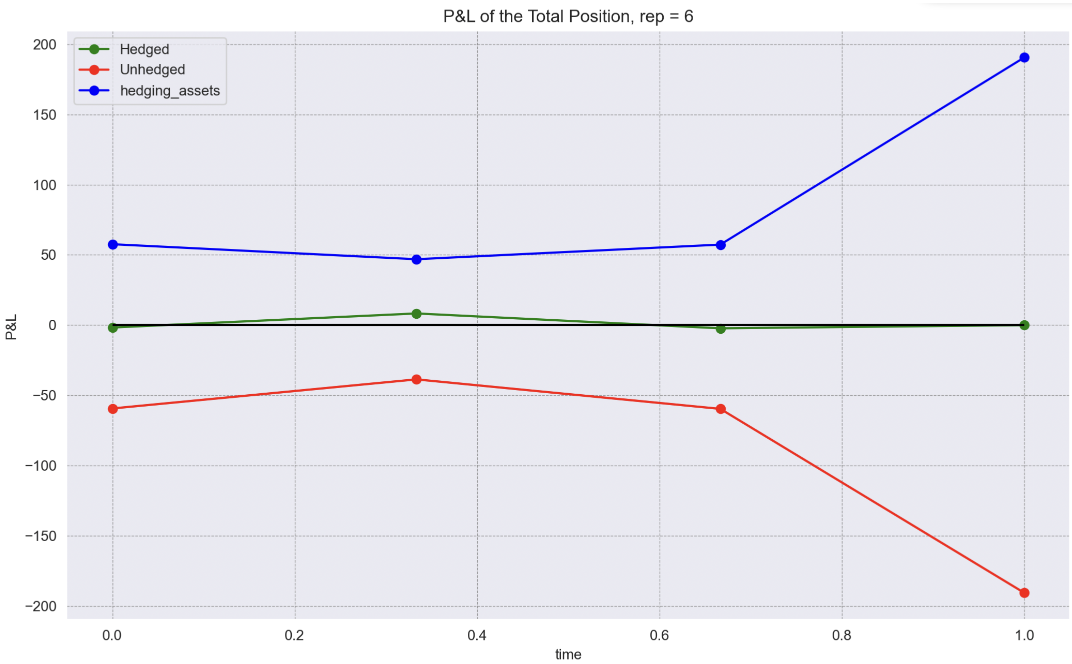


Figure 6.24: Evolution in time of a down-and-in call option which has come into existence and is exercised at maturity by its holder.

This option is used to illustrate the behaviour of the strategy across the three possible situations that may arise at maturity:

- the option has been activated and then exercised at maturity, resulting in a loss for the bank. However, if a stochastic optimization-based hedging strategy is adopted, it is possible to fully offset the short position at maturity T through a proper hedging portfolio, rebalanced until time $T-1$, as depicted in Figure 6.24. This is not surprising: when active, barrier options behave like standard vanilla options at maturity and previous sections have already demonstrated the good performance of stochastic optimization in such cases;

- the option comes into existence, but at maturity it is not exercised because the spot price does not exceed the strike price.

Figure 6.25: Evolution in time of a down-and-in call option which has come into existence but is not exercised at maturity by its holder.

Once again, the rebalanced portfolio is able to follow and replicate the target asset, resulting in an overall net-zero position for the hedged strategy at maturity. Figure 6.25 shows a Monte Carlo replication in which this situation happens: in particular, the option comes into existence at the second time instant.

- the option remains inactive since the barrier is never reached, resulting in a zero payoff. An example of such cases is depicted in Figure 6.26; the portfolio is hedging a short position that in fact never comes into effect. Clearly this cannot be known a priori, thus the strategy still manages to replicate the target asset behaviour and once again it does it successfully, leading to a total P&L equal to zero.

Figure 6.26: Evolution in time of a down-and-in call option which never becomes active.

Down-and-out put: numerical evidence

To numerically verify the solutions derived from the applied hedging strategy, a down-and-out put option was employed. Its main features are summarized in Table 6.14. The hedging portfolio is constructed using the initial capital obtained from the sale of the option. As its value evolves over time, it is rebalanced until the time step before maturity. Attention is now shifted to the last rebalancing step to illustrate the mechanics of the adopted strategy. The hedging portfolio at time $T-1$ before rebalancing is composed as shown in Table 6.15, which in fact is the same composition that comes from time $T-2$ after the adjustment. Imagining to be at $T-1$, by observing the current prices the value of the portfolio is determined and then used to finance any necessary rebalancing, resulting in the final portfolio holdings, which will no longer be adjusted and will be held until maturity. After

	stocks			strike	barrier
	MMM	STOXX50E	TESLA	300	180
initial price	242.92	102.05	107.19		
put strike	250	110	120		
call strike	230	100	100		

Table 6.14: Down-and-out put features: underlying (the highlighted one), risky hedging instruments, strike, barrier.

solving the problem using a single-period scenario tree, the optimizer returns the solution in Table 6.16.

MMM		STOXX50E		TSLA		CASH
0.34525		−0.25828		0		29.33589
PUT	CALL	PUT	CALL	PUT	CALL	
0.20770	−0.76797	0	−4.71034	0	0.0791	

Table 6.15: Portfolio composition at $T-1$ before rebalancing.

MMM		STOXX50E		TSLA		CASH
−0.43084		0		0		151.5571
PUT	CALL	PUT	CALL	PUT	CALL	
0.56915	0	0	−0.61901	0	0	

Table 6.16: Portfolio composition at $T-1$ after rebalancing, which also corresponds to the portfolio composition at maturity.

As can be seen, the final portfolio composition includes only the underlying asset, vanilla put options on it, call options on STOXX50E and cash. Comparing it to the previous portfolio, the adjustments involved, for example, selling a certain amount of the underlying asset, shifting the position from 0.34525 to −0.43084, buying $0.56915 - 0.20770 = 0.36145$ put options on it, selling all 0.07591 call options on TSLA, and so on. At this point, the process simply requires waiting until maturity, when the new market prices will be observed and the final portfolio value will be computed. In this specific real-world scenario, the barrier option exists and is exercised by its holder, resulting in a payoff equal to \$97.321. Then, having the following maturity values

$$p_{cash,T} = 1.0406$$
$$p_{MMM,T} = 202.67875$$
$$p_{put\ MMM,T} = 47.321$$
$$p_{call\ STOXX50E,T} = 0$$

the hedging portfolio at maturity will be

$$\begin{aligned} V_P^T(T) = &151.5571 \cdot 1.0406 - 0.43084 \cdot 202.67875 + 0.56915 \cdot 47.321 + \\ &- 0.61901 \cdot 0 = \$97.321 \end{aligned}$$

It becomes evident that the hedged position holds a final value of zero, while the unhedged will incur a loss equal to the option payoff. This example also provides numerical validation of the chosen hedging strategy.
Looking at the entire history of the strategy from the beginning of the hedging horizon in this specific real-world simulation, it can be noticed that the optimizer never selects certain instruments, such as TSLA stock, put options on it or put options on the STOXX50E. This highlights a key aspect of the strategy's mechanism: despite having access to a broad set of hedging assets, it only includes those deemed necessary to hedge the target asset at maturity. This is an encouraging result in terms of transaction costs, as it prevents the inclusion of redundant instruments in the portfolio and avoids trades that are not efficient for the final hedging objective.

Up-and-out put: practical application

This subsection aims to assess ex post the hedging strategy proposed by the stochastic optimization approach. The analysis is made through an up-and-out put option, whose characteristics are detailed in Table 6.17. Given that knock-out options may expire before maturity during the hedging horizon, resulting in a guaranteed zero payoff, the unhedged position benefits from preserving the whole sale price of the derivative, which remains unused. On the other hand, the hedged position employs this amount to build the hedging portfolio. Noting that once a barrier option ceases to exist it becomes certain that there is no longer a need to hedge future cash flows at maturity, the following strategy can be adopted:

1. rebalance the portfolio as long as the option remains active, ensuring that any potential exercise is properly hedged;

2. once market conditions confirm that the option ceases to exist, rebalancing stops and the portfolio is immediately liquidated. The resulting capital is then invested at the risk-free rate until maturity.

	stocks			strike	barrier
	MMM	STOXX50E	TESLA	300	330
initial price	242.92	102.05	107.19		
put strike	250	110	120		
call strike	230	100	90		

Table 6.17: Up-and-Out put features: underlying (the highlighted one), risky hedging instruments, strike, barrier.

This approach ensures that the portfolio is not unnecessarily rebalanced to replicate a derivative that will certainly yield a zero payoff, allowing a gain rather than a 0 payoff for the hedged position. A numerical example is presented below.
For the considered Monte Carlo replication, when observing the actual value of the underlying asset at the second time step, i.e. \$349.954, the option is deactivated since the barrier 330 is hit. The portfolio is composed as shown in Table 6.18, which is the result of the rebalancing at the previous time step. Instead of rebalancing it again, the portfolio is sold at the current time, yielding a total of \$21.1648. If this quantity is invested until maturity at the risk-free rate, it will become \$21.4476 (holdings and values are shown, respectively, in Table 6.18 and Table 6.19).

MMM		STOXX50E		TSLA		CASH
0		−0.001338		0		71.81847
PUT	CALL	PUT	CALL	PUT	CALL	
0.62418	−0.23796	0.02354	1.81408	−0.06428	−0.80561	

Table 6.18: Portfolio composition when the barrier option ceases to exist.

MMM		STOXX50E		TSLA		CASH
349.95397		41.07357		119.65852		1.02689
PUT	CALL	PUT	CALL	PUT	CALL	
0.31246	123.05254	67.49202	0.03616	3.84203	30.84648	

Table 6.19: Hedging assets values when the barrier option ceases to exist.

This represents a notable positive result, especially when compared to the zero payoff that would be obtained by rebalancing the portfolio one more time. Clearly, compared to what would be obtained from an unhedged strategy, i.e. the initial price of the derivative invested until maturity at the risk-free rate

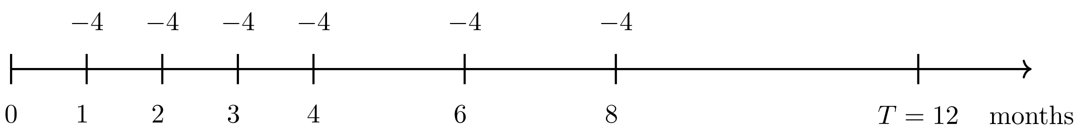
$$54.67197 \cdot \mathrm{e}^{0.0389(1-0.\bar{6})} = \$55.40212,$$

there is a loss. However, it is worth it, since the hedging strategy allows to cover a derivative whose existence at maturity is uncertain. It is important to note that the analyzed scenario is a specific one: the hedging strategy ensures risk minimization, as demonstrated in the previous sections.

6.10 WP derivatives

Unlike the derivatives analyzed so far, which may not be exercised at maturity, the first two worst-performance derivatives guarantee a certain loss for the bank. What remains uncertain is the actual cash outflow, but it will occur in every scenario. Therefore, this highlights the advantage of adopting a hedging strategy.
To facilitate numerical analysis, a simplified test setting has been implemented, ensuring that the structure and mechanism of the derivatives are preserved. In particular, worst-performance derivatives share the following features:

- AMZN, META and GOOGL are their underlyings, respectively with initial values 242.92, 102.05, 107.19;
- their maturity is one year;
- coupons are considered with the following occurrence:



- the issue price is 100;
- they have a percentage barrier of 0.95;
- two vanilla options are considered on each underlying:
  - three vanilla put options are taken into account, respectively with strike 250, 110 and 120;

- three vanilla call options are also considered, respectively with strike 230, 100 and 100.

  All of them expire in one year, matching the derivative maturity.

- a bank account which evolves at the risk-free rate is also considered as hedging asset;

- quarterly rebalancing is admitted, thanks to a branching factors vector with configuration [35,5,3] and similar. Note that a more complex configuration is required for these derivatives, in order to allow scenario trees to better capture their behaviour. Moreover, coupons are aggregated in these three time steps.

For each WP derivative, a self-financing with fixed initial wealth and without withdrawal possibility is chosen.
Starting from the simplest derivative, WP1, which is characterized by deterministic coupons, the comparison between the P&L of the hedged and unhedged positions is shown in the Figure 6.27. The fact that these derivatives have a piecewise payoff

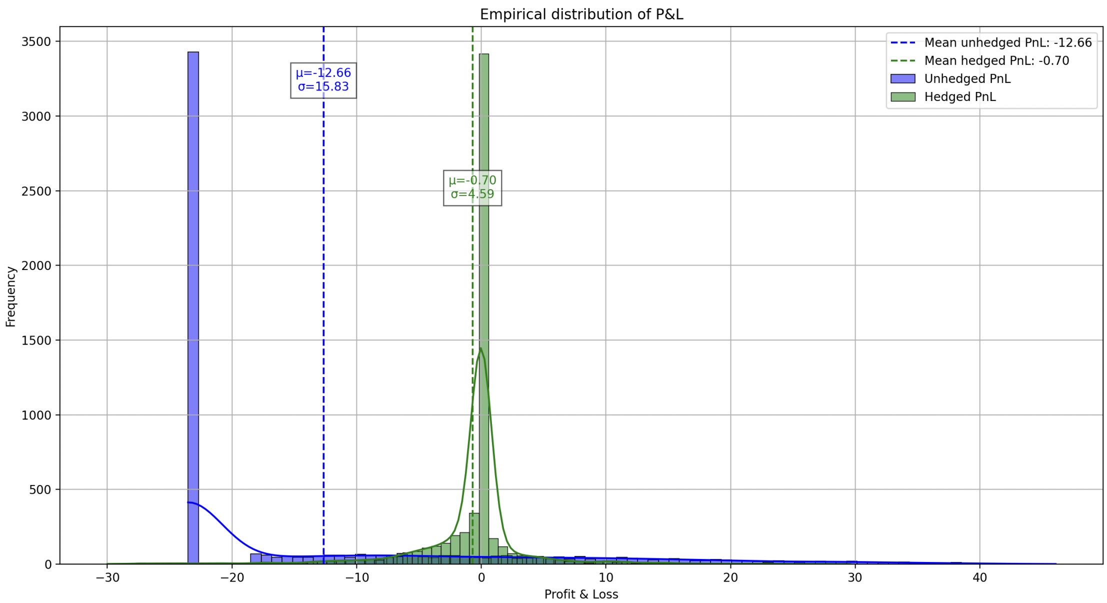


Figure 6.27: P&L empirical distributions of the unhedged (blue) and hedged (green) positions: WP1 case.

function is reflected in the distribution of the unhedged position. From the bank's perspective, the worst-case scenario occurs when, at maturity, the worst-performing underlying exceeds the barrier. In these cases, the payoff corresponds to the issue

price. By subtracting it from the initial premium (carried to maturity) and accounting for the guaranteed coupon payments, the final outcome is $-\$23.498$. This worst-case situation is represented by the high bar in the above histogram.
The limited-risk nature of this derivative from the holder point of view is evident: in 3500 over 6000 simulations, the highest possible payoff is obtained, in addition to the scheduled coupon payments over the investment horizon. Clearly, on the other hand, it is quite risky for the bank, fact that justifies the adoption of a hedging strategy. Its application leads to an excellent risk reduction, with an average P&L of -0.70 and a standard deviation of 4.59.
To verify that the actual cost of risk mitigation for this type of derivative is not excessive, an analysis of pricing via hedging is conducted. The results are shown in Figure 6.28. As can be observed, the prices required by a self-financing strategy are within reasonable bounds around the theoretical price. This highlights the reliability and practical applicability of the stochastic optimization approach, even when applied to worst-performance derivatives.

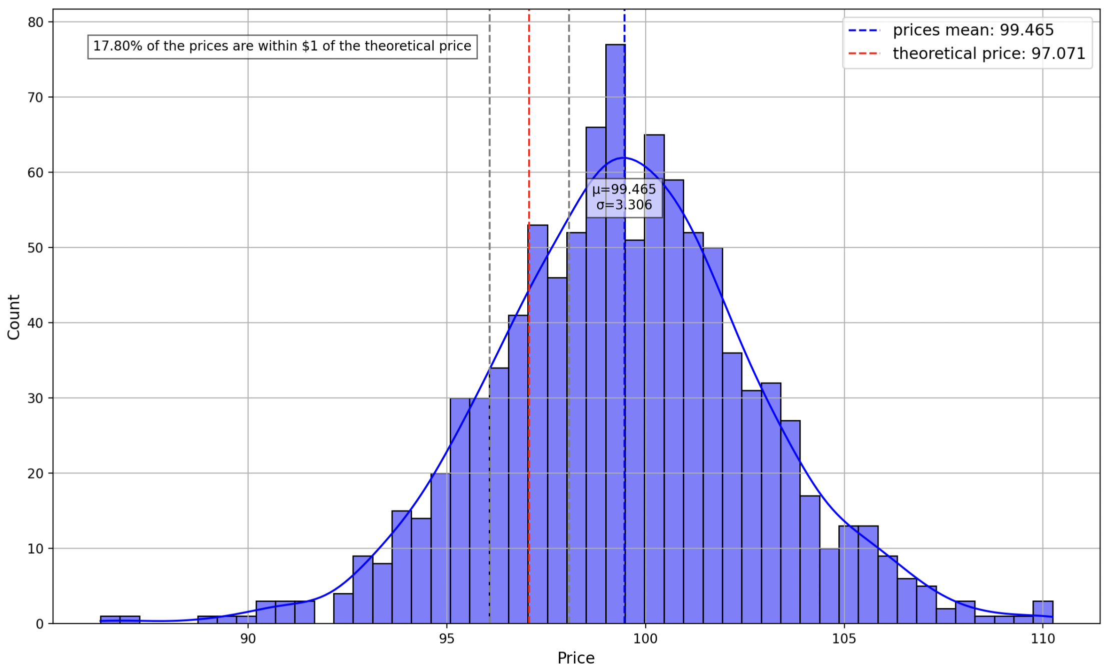


Figure 6.28: Initial wealth distribution W_0 chosen by the optimization solver, compared to the theoretical Monte Carlo price of WP1.

The focus now shifts to WP2, which still guarantees a loss at maturity, but adds an extra degree of uncertainty since coupons may not be paid (this depending on whether the worst-performing asset on coupon dates exceeds the digital barrier

of 75% or not). The more complex structure of WP2 results in longer computational time: adding a barrier check at each step to determine whether coupons will be paid increases the time required to complete the simulation. This effect is amplified in scenario trees with a complex configuration and higher number of scenarios.
Figure 6.29 is provided to better illustrate the cash flow that the bank has to sustain at maturity. As can be observed, in 100% of the cases the bank incurs a loss at maturity, which is important to be managed through a proper hedging strategy. These losses are not negligible, averaging around \$90. At least, there is a lower bound which avoids the risk of unlimited losses. At the same time, the variance of the unhedged position for this type of derivative is lower compared to the earlier analyzed options. This results from the fact that the maturity cash flow is determined as a percentage of the issue price: 100% in the worst case, but lower in more advantageous situations where is given by the worst-performance among underlying assets at maturity. This implies that losses are constrained within a defined range.

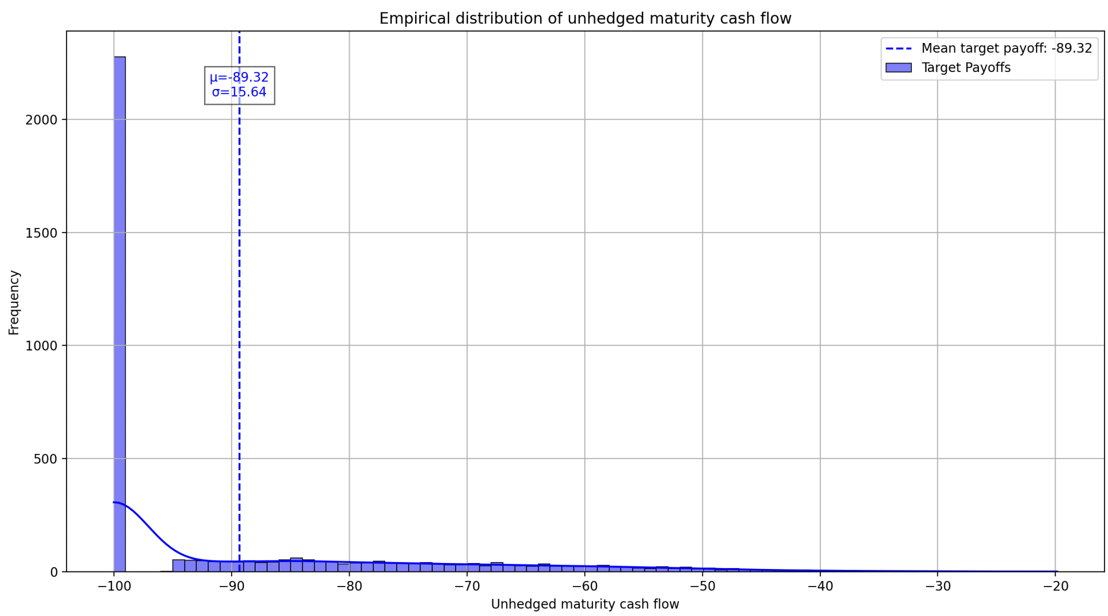


Figure 6.29: Distribution of the cash flow that the bank has to sustain at maturity in a short position on a WP2 derivative.

In this case as well, adopting a self-financing strategy without the possibility to withdraw money leads to a positive result, reducing the P&L variance from 20.56 to 7.46. A comparison between the unhedged and hedged position resulting from Monte Carlo replications is depicted in Figure 6.30.
The inclusion of coupons to obtain the overall P&L adds variability. In WP1, the

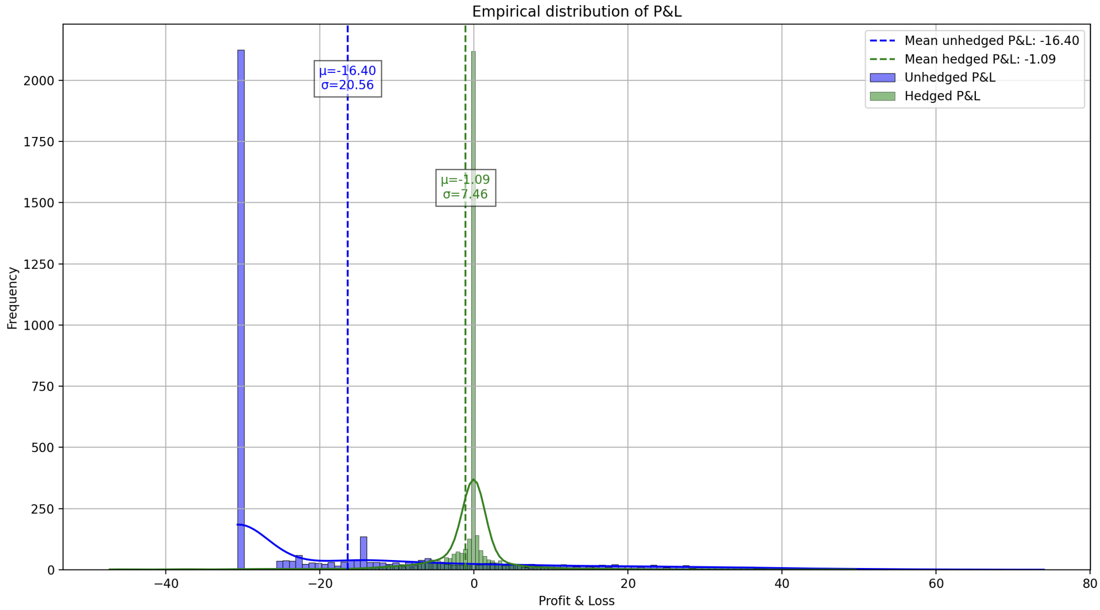


Figure 6.30: P&L empirical distributions of the unhedged (blue) and hedged (green) positions: WP2 case.

unhedged P&L is determined by simply subtracting the total coupons received and the payoff from the initial price: excluding the final outflow, the others are merely constants, thus the payoff distribution's standard deviation coincides with the one of the P&L. However, in WP2 the fact that coupon payments are not guaranteed introduces an additional source of variability, leading to a higher variance in the profit and loss distribution rather than in the final payoff distribution.

The last and most challenging worst-performance derivative is now analyzed. Its complexity is due not only to the stochastic nature of the coupon payments, but also to the potential early redemption, which deactivates the derivative before maturity. In summary, this derivative is subject to three main sources of uncertainty:

1. whether coupons will be paid or not;
2. whether the derivative still exists until maturity or not;
3. the final cash flow at maturity.

The discontinuity of the payoff function of this short position is shown in Figure 6.31. Here, the upper and lower bounds of the loss are clearly visible. Like WP2, it exhibits a negative region that corresponds to scenarios where the contract reaches maturity and the bank is required to fulfill payment obligations. However, there is a key difference from the previous worst-performance derivatives: a zero payoff can also arise, introducing another source of discontinuity. Even in this more challenging

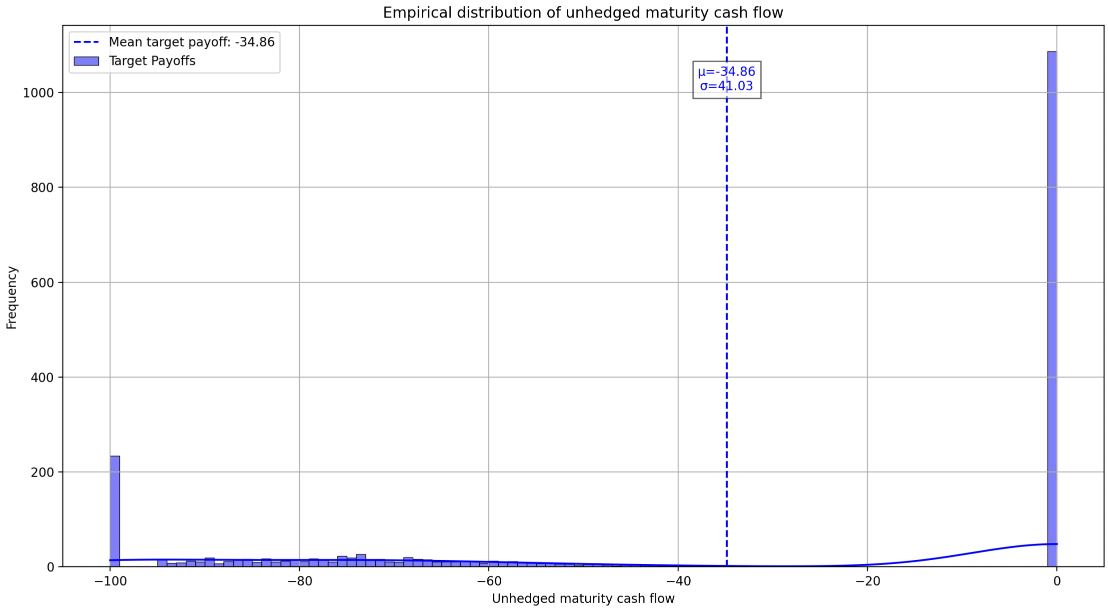


Figure 6.31: Distribution of the cash flow that the bank has to sustain at maturity in a short position on a WP3 derivative.

case, the implemented optimizer mitigates the key risk factors associated with WP3, once again confirming the power and robustness of stochastic optimization applied to the hedging framework. Figure 6.32 shows the benefits from a P&L perspective.

Since this derivative may cease to exist, as observed with out-barrier options, a more advantageous approach could be to stop rebalancing the portfolio when it becomes inactive and instead liquidate it and reinvest the obtained funds at the risk-free rate for the remaining time until maturity. This approach allows for a gain at maturity rather than a zero-value position resulting from replicating the target asset's value. Although the profit will be lower than the premium received at $t = 0$, hedging remains essential, even more than with standard barrier options. This arises not only from the potential liabilities associated with coupon payments during the hedging horizon (which are absent in standard barrier options, where the only cash flow to hedge is the final payoff), but also from the possibility of early redemption. This event entails the payment of a large amount of money (much greater than coupons) before maturity. For this reason, adopting a hedging strategy is worthwhile even in scenarios where the derivative may cease to exist in the future. Additionally, the unhedged position remains highly risky regardless of the contract's inactivity: in fact, if the early redemption takes place at time $t = \tau$,

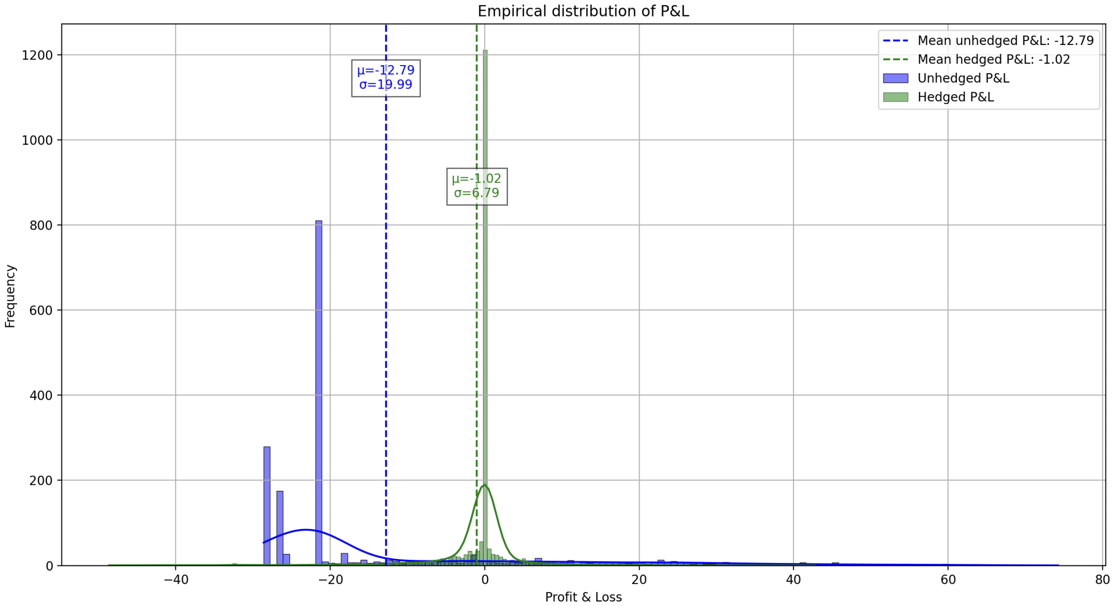


Figure 6.32: P&L empirical distributions of the unhedged (blue) and hedged (green) positions: WP3 case.

the overall P&L for the unhedged will be

$$\text{P\&L}_{unhedged} = \Pi(0)\mathrm{e}^{rT} - 0 - \sum_{t=0}^{\tau} l_t \cdot \mathrm{e}^{r(T-t)} \tag{6.5}$$

since $\Psi(T) = 0$. The quantity l_τ will also contain the early redemption amount. Considering that the latter is similar to the issue price, (6.5) proves to be surely negative. On the contrary,

$$\text{P\&L}_{hedged} = \text{he} = 0 \tag{6.6}$$

as all kinds of intermediate liabilities (the early redemption amount, too) are assumed to be fully covered by the self-financing strategy which starts with an initial wealth equal to $\Pi(0)$. As previously said, it is possible to interrupt the rebalancing and obtain a net profit at maturity by investing the capital gained from liquidating the last rebalanced portfolio.

One drawback of using stochastic optimization for this type of derivative is the computational time required. A solid branching factor configuration is necessary to obtain applicable results, but frequent checks for the three barriers (for coupons, for early redemption and, eventually, for the payoff), inevitably increase computational time.

Bibliography

[1] Michael Villaverde. «Hedging European and Barrier options using stochastic optimization». In: *Quantitative Finance* (2004), pp. 549–557 (cit. on pp. 23, 84).

[2] G. Amici, P. Brandimarte, F. Messeri, and P. Semeraro. *Multivariate Lévy Models: Calibration and Pricing.* July 2023. DOI: 10.48550/arXiv.2303.13346.

[3] John C. Hull. *Options, Futures and Other Derivatives.* Pearson, 2021.

[4] M. Capinski and T. Zastawniak. *Mathematics for Finance: An Introduction to Financial Engineering.* Springer, 2010.

[5] Intesa Sanpaolo. *Key Information Document: Standard Long Barrier Plus Worst of Certificates.* 30 March 2022.

[6] Intesa Sanpaolo. *Key Information Document: Standard Long Barrier Digital Worst of Certificates.* 6 April 2022.

[7] Intesa Sanpaolo. *Key Information Document: Standard Long Autocallable Barrier Digital Worst of Certificates with Memory Effect.* 28 October 2022.

[8] K. Høyland, M. Kaut, and S. W. Wallace. «A Heuristic for Moment-Matching Scenario Generation». In: *Computational Optimization and Applications* (2003). DOI: 10.1023/A:1021853807313.

[9] Roy Kouwenberg. «Scenario generation and stochastic programming models for asset liability management». In: *European Journal of Operational Research 134* (2001), pp. 279–292.

[10] K. Høyland and S. W. Wallace. «Generating Scenario Trees for Multistage Decision Problems». In: *Management Science* (2001), pp. 295–307. DOI: 10.1287/mnsc.47.2.295.9834.

[11] J. Gondzio, R. Kouwenbergb, and T. Vorst. «Hedging options under transaction costs and stochastic volatility». In: *Journal of Economic Dynamics Control* 27 (2003), pp. 1045–1068.