ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Deep Hedging the Volume-Weighted Average Price Risk in Order-Driven Markets

Master Thesis
Master of Science UZH ETH in Quantitative Finance

Michael Geiser
Student number: 22-738-645

November 28, 2025

Advisor: Prof. Dr. Josef Teichmann
Department of Mathematics, ETH Zürich

Contents

Abstract

We study volume-weighted average price execution risk in order-driven markets within the Deep Hedging framework. A Santa Fe limit order book model, calibrated to LOBSTER intraday data for four NASDAQ stocks, generates market scenarios. The agent trades through a liquidity provider that returns execution prices for market orders and exogenous fill rules for limit orders, ignoring market impact. Policies are represented by a system of neural networks that map current market features and previous actions to a market order size and a vector of limit order placements. Training minimizes a convex risk measure of terminal wealth using TensorFlow on simulated paths. The learned policies are stock-specific and economically interpretable: tight-spread books use early market orders and shallow limit order posting before shifting passive; spread-out books rely on purely passive deep placement; intermediate structures favor mid-level passive accumulation. Deep Hedging thus emerges as a viable approach in order-driven markets, with microstructure conditions shaping optimal market and limit order allocation.

Chapter 1

Introduction

We study the optimal execution policy for a large order that targets the volume-weighted average price (VWAP) over an intraday horizon, a central problem for trading desks. While individual trading strategies are typically proprietary, market consensus acknowledges that a substantial proportion of institutional flow is benchmarked to VWAP, and managing associated execution risk is critical to performance.

The problem is particularly challenging in limit order book (LOB) markets, which account for a large share of electronic trading activity among professional participants. The added complexity stems from a high-dimensional action space (a time-size-price grid) and the intricate dynamics of state-dependent order arrivals.

We adopt the perspective of an agent submitting market and limit orders to a liquidity provider (execution engine). The provider does not route orders to the market but instead returns execution prices for market orders and exogenous fill rules for limit orders. This abstraction reflects the setting of an institutional client interacting with a trading house. It also bypasses explicit price–time priority and queue-position modeling by using a simpler fill rule (e.g., fills at touched price levels).

We show that the Deep Hedging (DH) framework of Buehler et al. [3] is both effective and computationally tractable for managing VWAP risk in an LOB setting. We train a policy neural network to learn optimal allocations between market and limit orders that hedge a VWAP-linked liability, accounting for transaction costs and stochastic order fills.

The remainder of this chapter introduces the VWAP risk-hedging problem, reviews recent developments in the literature, and outlines our objectives and methodology.

1.1 VWAP Risk-Hedging Problem

When an agent must execute a large buy or sell order over one trading day, performance is typically measured against the VWAP, which reflects the average cost of market participation during that period. Concrete scenarios include fulfilling execution mandates with VWAP objectives or rebalancing portfolios after the expiration of cash-settled derivatives linked to VWAP.

We work in an order-driven, continuous double-auction market where prices and volumes emerge from two principal order types:

- **Limit order** (LO): instruction to buy or sell a given quantity at a specific price, which executes only if a counterparty agrees to trade at that price.
- **Market order** (MO): instruction to buy or sell a given quantity immediately at the best available prices.

MOs are matched with available LOs according to a *price-time priority rule*: better prices execute first, and within a price level, earlier orders have priority. Outstanding LOs may be partially or fully canceled at any time.

These order types entail a trade-off: MOs guarantee immediate execution but pay at least half the bid–ask spread (the difference between the best current sell and buy prices), while LOs can save at least half the spread but introduce execution risk.

In our framework, the agent transacts exclusively with a liquidity provider. The provider does not place orders directly into the LOB; instead, he provides prices and fills orders based on exogenous rules derived from the current market state and its evolution. There are no additional frictions, such as fees, and trades executed through the provider do not affect the market (i.e., *no market impact*).

Remark 1.1. VWAP mandates are usually enforced as same-side execution: no intraday round-trip trading (e.g., selling during a buy program).

We focus on the buy side; the sell case is symmetric. The agent aims to purchase $Q_0 > 0$ units over the interval $[0, T]$, corresponding to one trading day, using a finite number of decisions at times $0 \le t_0 < t_1 < \cdots < t_{N-1} < T$. At each time t_n, the agent can submit buying MOs and place buying LOs at $L \in \mathbb{N}$ price levels $B_n^{(1)}, \ldots, B_n^{(L)}$.

The trading policy is represented by

$$\delta = \{\boldsymbol{\delta}_n\}_{n=0}^{N-1}, \qquad \boldsymbol{\delta}_n = (\varphi_n, \vartheta_n^{(1)}, \ldots, \vartheta_n^{(L)}) \in \mathbb{R}_{\ge 0}^{L+1}, \tag{1.1}$$

where φ_n is the MO quantity executed at an average price $A_n^{(0)}(\varphi_n)$ depending on φ_n, and $\vartheta_n^{(\ell)}$ is the LO quantity posted at price $B_n^{(\ell)}$, $\ell \in \{1, \ldots, L\}$.

The policy generates an executed quantity Q^δ and an average price P^δ given by

$$\begin{aligned} Q^\delta &= \sum_{n=0}^{N-1} \left(m_n(\varphi_n) + \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)}) \right), \\ P^\delta &= \begin{cases} \dfrac{\sum_{n=0}^{N-1} \left(m_n(\varphi_n) A_n^{(0)}(\varphi_n) + \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)}) B_{n-1}^{(\ell)} \right)}{Q^\delta}, & \text{if } Q^\delta > 0, \\ 0, & \text{if } Q^\delta = 0, \end{cases} \end{aligned} \tag{1.2}$$

where $m_n : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ caps the MO to not exceed the available LOB liquidity, and $r_n^{(\ell)} : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ denotes the LO filled over $(t_{n-1}, t_n]$.

VWAP is computed from MOs, excluding the agent's trades. Denote by $\{(Q_m, P_m)\}_{m=1}^{M_T}$ the quantities and execution prices of these MOs over the interval $[0, T]$, with $\sum_{m=1}^{M_T} Q_m > 0$. Then,

$$\mathrm{VWAP} = \frac{\sum_{m=1}^{M_T} Q_m P_m}{\sum_{m=1}^{M_T} Q_m}. \tag{1.3}$$

The agent's terminal wealth relative to VWAP is

$$W^{\delta}_{\mathrm{VWAP}} = Q_0\big(\mathrm{VWAP} - S_T\big) \; - \; Q^{\delta}\big(P^{\delta} - S_T\big), \tag{1.4}$$

where S_T is the terminal mid-price. Equation (1.4) can be interpreted as the agent carrying out two hypothetical transactions valued at the closing mid-price: in the first, Q_0 units are sold at VWAP, and in the second, Q^{δ} units are purchased at P^{δ}. The resulting wealth reflects the combined mark-to-market profit or loss of these trades relative to the terminal mid-price S_T.

The agent chooses δ to hedge the downside of $W^{\delta}_{\mathrm{VWAP}}$ (i.e., extreme negative losses). On a filtered probability space $(\Omega, \mathbb{F}, \mathbb{P})$ and for the set of admissible policies $\mathcal{H}$, we solve

$$\pi = \inf_{\delta \in \mathcal{H}} \; \rho\big(W^{\delta}_{\mathrm{VWAP}}\big)\,, \tag{1.5}$$

where $\rho : \mathcal{X} \to \mathbb{R}$ is a convex risk measure (e.g., Expected Shortfall), and $\mathcal{X}$ is the set of all real-valued random variables over Ω.

Finding an optimal policy by solving (1.5) is difficult for two reasons. First, the action space is high-dimensional: the agent must decide the timing, sizes, and LO price levels. Second, LO fills are unknown: they depend on the LOB dynamics of the next sub-interval, which introduces a stochastic component to the agent's inventory evolution.

1.2 Related Work

Machine learning approaches to LOBs have focused primarily on optimal trade execution, with related VWAP studies typically targeting slippage minimization rather than *risk*. One strand learns execution schedules from LOB snapshots while abstracting away queue dynamics. Lin and Beling [9] trains an end-to-end scheduler that maps aggregated snapshots, remaining inventory, and time to the quantity executed at each decision interval, assuming immediate taking against top-of-book depth. Training uses a sparse, end-of-episode reward, and performance is reported via mean slippage and gain–loss ratio. This design targets the mean episode-level cost.

A second strand operates in simulated LOBs with discrete action sets and reward structures centered on slippage. Karpe et al. [7] use features such as spread, imbalance, short-horizon returns, and remaining time/quantity; actions are finite deviations from a time-weighted average price (TWAP) schedule with finite placement choices (MOs or shallow LOs). Li et al. [8] propose a hierarchical VWAP framework (Macro/Meta/Micro) to minimize VWAP slippage: the Macro module predicts the intraday volume profile and allocates slices; the Meta module sets subgoals (horizon and child size); and the Micro module interacts with the LOB, choosing to place at best bid/ask or wait, with unfilled quantities converted to MOs at the subgoal deadline.

A third line increases environment fidelity with market-by-order simulators but typically retains mean-cost objectives. Byun et al. [4] train an agent with actions (MO, new LO, modify LO, pass) and placements restricted to the top five levels per side in a price–time-priority simulator. Genet [5] use recurrent networks that ingest market features over a lookback window and, at each step, output the fraction of the parent order to execute, aiming to minimize VWAP slippage.

1.3 Objective and Methodology Overview

Most prior research optimizes mean VWAP slippage, uses discrete or top price levels for orders, or applies value-function approximation with designed rewards. Explicit VWAP risk-hedging via gradient-based optimization over the full order book depth remains comparatively underexplored.

Our approach departs from previous work along three axes:

- We optimize a coherent risk measure of terminal wealth instead of average slippage.
- We adopt a differentiable-programming framework that enables gradient-based optimization and backpropagation through a piecewise-differentiable loss, thereby avoiding reward-based value-function approximation.
- We broaden the action space to cover allocations across the full depth of the order book, rather than restricting decisions to top-of-book trading.

We stress that our approach can be adapted to different LOB market models, providing a ready-to-implement solution with industry applications.

The methodology unfolds as follows. We first develop a simulator that captures the key execution mechanisms of a LOB market. The environment is defined by a LOB representation model, state-update rules, and a path-simulation procedure (Section 2.1). Within this setting, we formulate the agent's optimization problem: selecting MOs and LOs across multiple price levels to buy a target quantity of an asset, with the average execution price compared against VWAP (Section 2.2).

We then embed this problem in the DH framework. A policy network maps market features and past actions to new MO and LO allocations (Section 2.3). Training is fully end-to-end via stochastic gradient descent (SGD), yielding policies that directly minimize a risk-based objective (Section 2.4). The neural network architecture is described next (Section 2.5).

Finally, we validate the approach using real market data. We estimate model parameters for the LOB simulator across different market regimes and construct the corresponding training datasets (Section 3.1). We train and evaluate the learned VWAP risk-hedging policies, examining how their structure varies with underlying model parameters (Section 3.2), present the main findings (Section 4.1), and discuss directions for future research (Section 4.2).

Chapter 2

Methodology

In this chapter, we develop the theoretical justification and practical implementation for solving the VWAP risk hedging problem in a realistic LOB market. We begin by specifying a parametric LOB model that captures key microstructure features. These include queue dynamics, order flow, and price updates. Next, we integrate the agent's trading policy into the market model and formulate the agent's optimization problem. This approach naturally leads to the DH framework, which enables the use of neural networks to approximate the optimal trading policy. Finally, we specify the training procedure and the network architecture for our numerical experiments.

2.1 Model Construction

In this section, we introduce a parametric LOB model. We use the *Santa Fe* queueing system model described in detail in [2]. The general representation of LOBs, the state-update rules, and the Santa Fe–based simulation algorithm are described in [1].

2.1.1 Limit Order Book Representation

We assume that the market starts at time 0 in an observable state and evolves stochastically until time $T > 0$, denoting the end of the trading session.

At any time $t \in [0, T]$, each side of the LOB is described by $K \in \mathbb{N}$ relative prices, indexed 1 to K ticks away from the best *opposite* quote. Absolute prices lie on the grid $\epsilon\mathbb{N}$, where $\epsilon > 0$ is the tick size, representing the minimum price increase allowed.

Let $\mathbf{a}_t = (\mathrm{a}_t^{(1)}, \dots, \mathrm{a}_t^{(K)})$ denote the **ask queues** at time t, where $\mathrm{a}_t^{(k)} \geq 0$ is the volume offered k ticks from the best opposite quote. Likewise, $\mathbf{b}_t = (\mathrm{b}_t^{(1)}, \dots, \mathrm{b}_t^{(K)})$ denotes the **bid queues**, where $\mathrm{b}_t^{(k)} \geq 0$ is the volume bid k ticks away from the best opposite quote.

To prevent queue depletion, we impose constant bounds outside the $2K$ grid: whenever levels shift and a new outermost level appears, its queue is initialized to $c^{\infty} > 0$.

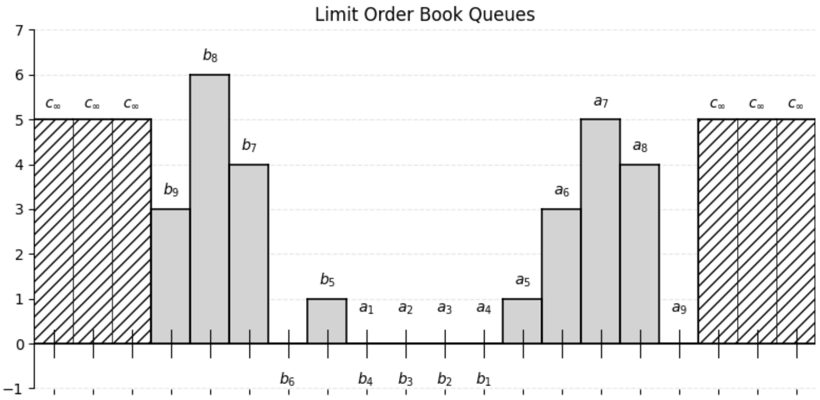

Figure 2.1: In this example, the ask and bid queues are $\mathbf{a}_t = (0,0,0,0,1,3,5,4,0)$ and $\mathbf{b}_t = (0,0,0,0,1,0,4,6,3)$, respectively. The number of visible relative prices is $K = 9$, and the outside bound is $c^\infty = 5$.

For convenience, we define the cumulative depth and its generalized inverse:

$$\Sigma_k(\mathbf{y}) = \sum_{i=1}^{k} y_i, \qquad \Sigma^{-1}(q,\mathbf{y}) = \inf_{i\in\{1,\dots,K\}} \{i : \Sigma_i(\mathbf{y}) > q\}, \tag{2.1}$$

where $k \in \{1,\dots,K\}$, $\mathbf{y} \in \mathbb{R}^K_{\geq 0}$, and $q \geq 0$. If $q \geq \Sigma_K(\mathbf{y})$, we set $\Sigma^{-1}(q,\mathbf{y}) = K$. We impose $\Sigma_0(\mathbf{y}) = 0$ for any $\mathbf{y}$.

The index corresponding to the first non-zero entry on the bid or ask queue is given by

$$i_t = \Sigma^{-1}(0,\mathbf{a}_t) = \Sigma^{-1}(0,\mathbf{b}_t). \tag{2.2}$$

The boundary conditions ensure that $i_t < \infty$ for all $t \in [0,T]$.

The **best ask price** $A_t \in \epsilon\mathbb{N}$ is determined by the price associated with the first non-zero ask queue. Similarly, the **best bid price** $B_t \in \epsilon\mathbb{N}$ is determined by the price corresponding to the first non-zero bid queue.

The **mid-price** S_t and the **bid-ask spread** ε_t are given by

$$S_t = \tfrac{1}{2}(A_t + B_t), \qquad \varepsilon_t = A_t - B_t = i_t\,\epsilon. \tag{2.3}$$

We use the notation

$$\mathbf{L}_t = (S_t, \varepsilon_t, \mathbf{a}_t, \mathbf{b}_t) \tag{2.4}$$

to describe the **LOB state** at time t.

Remark 2.1. By modeling aggregated queues, this LOB representation ignores the typical matching mechanism, *price–time priority*, and imposes only *price priority*. Within a price level, executions are assumed to split pro rata by LO size, thereby avoiding the need to track individual LOs and submission times and reducing modeling complexity.

2.1.2 State Update

The dynamics of the LOB state are driven by market participants' actions. Each action is represented as a discrete event: the arrival of buy or sell MOs, the arrival of buy or sell LOs, and the full or partial cancellation of existing buy or sell LOs.

Given a current state $\mathbf{L}_t$, we apply the following rules to update the LOB state when a new event occurs.

Buy Market Order. Assume a buy MO of size $q > 0$ arrives at time $t + \Delta t$, and let $i = \Sigma^{-1}(q, \mathbf{a}_t) - i_t$. Observe that $i > 0$ only if the order size fully consumes the best available liquidity. The state then evolves according to

$$S_{t+\Delta t} = S_t + \tfrac{1}{2} i\,\epsilon, \tag{2.5}$$

$$\varepsilon_{t+\Delta t} = \varepsilon_t + i\,\epsilon, \tag{2.6}$$

$$\mathbf{a}_{t+\Delta t} = \big(\min\{\mathrm{a}_t^{(1)}, (\Sigma_1(\mathbf{a}_t) - q)_+\}, \dots, \min\{\mathrm{a}_t^{(K)}, (\Sigma_K(\mathbf{a}_t) - q)_+\}\big), \tag{2.7}$$

$$\mathbf{b}_{t+\Delta t} = \big(\underbrace{0, \dots, 0}_{i \text{ times}}, \mathrm{b}_t^{(1)}, \dots, \mathrm{b}_t^{(K-i)}\big). \tag{2.8}$$

Buy Limit Order. If a buy LO of size $q > 0$ arrives at relative price $k \in \{1, \dots, K\}$, set $i = (i_t - k)_+$. Note that $i > 0$ only when the order is priced within the bid-ask spread. The state is updated as

$$S_{t+\Delta t} = S_t + \tfrac{1}{2} i\,\epsilon, \tag{2.9}$$

$$\varepsilon_{t+\Delta t} = \varepsilon_t - i\,\epsilon, \tag{2.10}$$

$$\mathbf{a}_{t+\Delta t} = \big(\mathrm{a}_t^{(1+i)}, \dots, \mathrm{a}_t^{(K)}, \underbrace{c^\infty, \dots, c^\infty}_{i \text{ times}}\big), \tag{2.11}$$

$$\mathbf{b}_{t+\Delta t} = \big(\mathrm{b}_t^{(1)}, \dots, \mathrm{b}_t^{(k-1)}, \mathrm{b}_t^{(k)} + q, \mathrm{b}_t^{(k+1)}, \dots, \mathrm{b}_t^{(K)}\big). \tag{2.12}$$

Cancellation of Buy Limit Order. Suppose $q > 0$ units of a buy LO at relative price k are canceled. Let $q' = \min\{q, \mathrm{b}_t^{(k)}\}$, and define $i = \Sigma^{-1}(q', \mathbf{b}_t) - i_t$ if $k = i_t$ and $i = 0$ otherwise. Observe that $i > 0$ whenever the best available liquidity is fully canceled. The state evolves to

$$S_{t+\Delta t} = S_t - \tfrac{1}{2} i\,\epsilon, \tag{2.13}$$

$$\varepsilon_{t+\Delta t} = \varepsilon_t + i\,\epsilon, \tag{2.14}$$

$$\mathbf{a}_{t+\Delta t} = \big(\underbrace{0, \dots, 0}_{i \text{ times}}, \mathrm{a}_t^{(1)}, \dots, \mathrm{a}_t^{(K-i)}\big), \tag{2.15}$$

$$\mathbf{b}_{t+\Delta t} = \big(\mathrm{b}_t^{(1)}, \dots, \mathrm{b}_t^{(k-1)}, \mathrm{b}_t^{(k)} - q', \mathrm{b}_t^{(k+1)}, \dots, \mathrm{b}_t^{(K)}\big). \tag{2.16}$$

The update formulas for sell MOs, sell LOs, and cancellations of sell LOs mirror the previous rules, with all mid-price adjustments occurring in the opposite direction. For example, a sell MO that consumes i bid levels pushes the mid-price downward to $S_t - \frac{1}{2} i\,\epsilon$.

2.1.3 Market Simulation

We adopt the Santa Fe model to simulate event times and event volumes: *times* are modeled by (conditionally) independent Poisson processes; *volumes* are sampled from independent random variables (r.v.s):

- **Arrival of market orders:** buy and sell arrivals are independent Poisson processes with intensity $\gamma > 0$ per side (aggregate rate 2γ); volumes are independent r.v.s with law $\mathcal{V}^{\mathrm{M}}$.
- **Arrival of limit orders at price level k:** for $k \in \{1, \dots, K\}$, buy and sell arrivals are independent Poisson processes with intensity $\lambda^{(k)} > 0$ per side (aggregate rate $2\lambda^{(k)}$); volumes are independent r.v.s with law $\mathcal{V}^{\mathrm{L}}$.
- **Cancellation of limit orders at price level k:** conditional on the current state, cancellations are Poisson processes with intensity $\rho^{(k)}\mathrm{b}_t^{(k)}$ (buy side) and $\rho^{(k)}\mathrm{a}_t^{(k)}$ (sell side), with $\rho^{(k)} > 0$; volumes are independent r.v.s with law $\mathcal{V}^{\mathrm{C}}$.

Here, γ and $\lambda^{(k)}$ denote arrival intensities per unit time, whereas $\rho^{(k)}$ is a parameter per unit time per unit volume; hence $\rho^{(k)}\mathrm{b}_t^{(k)}$ and $\rho^{(k)}\mathrm{a}_t^{(k)}$ are cancellation intensities per unit time.

Remark 2.2. Assuming that cancellation intensities at price level k are proportional to available volume implies that each LO has an exponentially distributed lifetime with parameter $\rho^{(k)}$.

To streamline the simulation of the LOB, we define the following **aggregate intensities**:

$$\lambda = \sum_{k=1}^{K} \lambda^{(k)}, \qquad \rho_t^{\mathrm{b}} = \sum_{k=1}^{K} \rho^{(k)}\mathrm{b}_t^{(k)}, \qquad \rho_t^{\mathrm{a}} = \sum_{k=1}^{K} \rho^{(k)}\mathrm{a}_t^{(k)}, \tag{2.17}$$

where λ is constant and ρ_t^a, ρ_t^b are state dependent (piecewise constant between events). The **total event intensity** is

$$\Lambda_t = 2(\gamma + \lambda) + \rho_t^{\mathrm{b}} + \rho_t^{\mathrm{a}}. \tag{2.18}$$

Remark 2.3. By superposition of independent Poisson processes, the aggregate intensity of each event category equals the sum of its subtypes' intensities (e.g., arriving LOs across all levels have an aggregate rate of 2λ). The total intensity Λ_t governs the waiting time to the next event regardless of event type.

To simulate the evolution of the LOB over a given time interval, we use **Algorithm 1**: at each step we draw a waiting time $\Delta t \sim \mathrm{Exp}(\Lambda_t)$, an event type, an order size, and a relative price $k \in \{1, \dots, K\}$, if applicable. Event types comprise buy/sell MOs, buy/sell LOs, and cancellations (CXL) of buy/sell LOs. The event type is selected with probability proportional to its component intensity (that is, γ/Λ_t per side for MOs, λ/Λ_t per side for LOs, $\rho_t^{\mathrm{b}}/\Lambda_t$ and $\rho_t^{\mathrm{a}}/\Lambda_t$ for buy/sell LO cancellations). For LOs arrivals, we sample the relative price k with weights $\lambda^{(k)}/\lambda$, while for cancellations, we sample with weights $\rho^{(k)}\mathrm{b}_t^{(k)}/\rho_t^{\mathrm{b}}$ (buy side) or $\rho^{(k)}\mathrm{a}_t^{(k)}/\rho_t^{\mathrm{a}}$ (sell side). After sampling, the LOB state is updated according to the rules explained in Subsection 2.1.2.

Algorithm 1: Simulate LOB path

```
inputs: order intensities $\gamma$, $\{\lambda^{(k)}\}_{k=1}^{K}$, $\{\rho^{(k)}\}_{k=1}^{K}$; volume laws $\mathcal{V}^{\mathrm{M}}$, $\mathcal{V}^{\mathrm{L}}$, $\mathcal{V}^{\mathrm{C}}$;
    start/end times $s$, $t$; initial state $\mathbf{L}_s = (S_s, \varepsilon_s, \mathbf{a}_s, \mathbf{b}_s)$
initialize: $\mathbf{L} \leftarrow \mathbf{L}_s$, $\tau \leftarrow s$, $\lambda \leftarrow \sum_{k=1}^{K} \lambda^{(k)}$
while $\tau \le t$ do
    Update cancellation intensities:
        $$\rho^{\mathrm{b}}_\tau \leftarrow \sum_{k=1}^{K} \rho^{(k)}\, \mathrm{b}^{(k)}_\tau, \quad \rho^{\mathrm{a}}_\tau \leftarrow \sum_{k=1}^{K} \rho^{(k)}\, \mathrm{a}^{(k)}_\tau$$
    Update total intensity: $\Lambda_\tau \leftarrow 2(\gamma + \lambda) + \rho^{\mathrm{b}}_\tau + \rho^{\mathrm{a}}_\tau$
    Simulate next-event time: draw $\Delta\tau \sim \mathrm{Exp}(\Lambda_\tau)$
    if $\tau + \Delta\tau > t$ then break while loop
    Simulate event type: draw $\mathtt{e} \sim \mathrm{Cat}\Big(\frac{\gamma}{\Lambda_\tau}, \frac{\gamma}{\Lambda_\tau}, \frac{\lambda}{\Lambda_\tau}, \frac{\lambda}{\Lambda_\tau}, \frac{\rho^{\mathrm{a}}_\tau}{\Lambda_\tau}, \frac{\rho^{\mathrm{b}}_\tau}{\Lambda_\tau}\Big)$
    Simulate quantity:
        if $\mathtt{e} \in \{\text{sell MO}, \text{ buy MO}\}$ then draw $\mathtt{q} \sim \mathcal{V}^{\mathrm{M}}$
        else if $\mathtt{e} \in \{\text{sell LO}, \text{ buy LO}\}$ then draw $\mathtt{q} \sim \mathcal{V}^{\mathrm{L}}$
        else if $\mathtt{e} \in \{\text{CXL sell LO}, \text{ CXL buy LO}\}$ then draw $\mathtt{q} \sim \mathcal{V}^{\mathrm{C}}$
        end if
    Simulate relative price:
        if $\mathtt{e} \in \{\text{sell LO}, \text{ buy LO}\}$ then draw $\mathtt{k} \sim \mathrm{Cat}\Big(\frac{\lambda^{(1)}}{\lambda}, \dots, \frac{\lambda^{(K)}}{\lambda}\Big)$
        else if $\mathtt{e} = \text{CXL sell LO}$ then draw $\mathtt{k} \sim \mathrm{Cat}\Big(\frac{\rho^{(1)}\mathrm{a}^{(1)}_\tau}{\rho^{\mathrm{a}}_\tau}, \dots, \frac{\rho^{(K)}\mathrm{a}^{(K)}_\tau}{\rho^{\mathrm{a}}_\tau}\Big)$
        else if $\mathtt{e} = \text{CXL buy LO}$ then draw $\mathtt{k} \sim \mathrm{Cat}\Big(\frac{\rho^{(1)}\mathrm{b}^{(1)}_\tau}{\rho^{\mathrm{b}}_\tau}, \dots, \frac{\rho^{(K)}\mathrm{b}^{(K)}_\tau}{\rho^{\mathrm{b}}_\tau}\Big)$
        end if
    Update time: $\tau \leftarrow \tau + \Delta\tau$
    Update state: $\mathbf{L} \leftarrow \mathbf{L} + (\mathtt{e}, \mathtt{q}, \mathtt{k})$
end while
output: $\mathbf{L}$
```

2.2 Execution Problem

In this section, we incorporate the agent's trading policy into the market model and define the corresponding optimization problem. At each decision time, the policy specifies the MOs and LOs to submit, which yields a policy-dependent random wealth. The objective is to choose a policy that minimizes a convex risk measure applied to this wealth. The formulation is model-free; it applies to any market model or dataset that supplies the requisite observables.

2.2.1 VWAP Policy Incorporation

We start by partitioning the trading horizon $[0, T]$ into $N \in \mathbb{N}$ sub-intervals specified by the sequence

$$0 = t_0 < t_1 < \cdots < t_N = T, \tag{2.19}$$

where t_n denotes the start of each sub-interval, with $n \in \{0, \dots, N-1\}$. We work on a filtered probability space $(\Omega, \mathbb{F}, \mathbb{P})$ with $\mathbb{F} = \{\mathcal{F}_n\}_{n=0}^N$. Let $I = \{\mathbf{I}_n\}_{n=0}^N$, $\mathbf{I}_n \in \mathbb{R}^r$, denote the stream of market information, and set $\mathcal{F}_n = \sigma(\mathbf{I}_0, \dots, \mathbf{I}_n)$. Finally, let $\mathcal{X} = \{X : \Omega \mapsto \mathbb{R}\}$ bet the set of all real-valued r.v.s over Ω.

If we take snapshots of the process $\mathrm{L} = \{\mathbf{L}_t\}_{t\in[0,T]}$ at each time t_n, we obtain a time-discrete, $\mathbb{F}$-adapted process $\widetilde{\mathrm{L}} = \{\widetilde{\mathbf{L}}_n\}_{n=0}^N$ by setting

$$\widetilde{\mathbf{L}}_n := (\widetilde{S}_n, \widetilde{\varepsilon}_n, \widetilde{\mathbf{a}}_n, \widetilde{\mathbf{b}}_n) = (S_{t_n}, \varepsilon_{t_n}, \mathbf{a}_{t_n}, \mathbf{b}_{t_n}), \qquad n = 1, \dots, N. \tag{2.20}$$

Now, fix $L \in \mathbb{N}$ such that $L \leq K$, and consider the $(L+1)$-dimensional, $\mathbb{F}$-adapted stochastic process

$$(\varphi, \vartheta) = \{(\varphi_n, \boldsymbol{\vartheta}_n)\}_{n=0}^{N-1}, \tag{2.21}$$

where $(\varphi_n, \boldsymbol{\vartheta}_n) = (\varphi_n, \vartheta_n^{(1)}, \dots, \vartheta_n^{(L)})$. Denote the space of such processes by $\mathcal{H}$.

The process (2.21) represents the **agent's trading policy**: at each time t_n, the agent chooses actions consisting of:

- a buy MO with size $\varphi_n \geq 0$;
- L buy LOs, each with a relative price $\ell \in \{1, \dots, L\}$ and a size $\vartheta_n^{(\ell)} \geq 0$.

Assuming that the agent starts flat (no prior positions), we set $(\varphi_{-1}, \boldsymbol{\vartheta}_{-1}) = \mathbf{0}$.

At time t_n, the **agent's execution quantity** $Q_n^{(\varphi,\vartheta)}$ is the number of units executed via MOs and filled LOs during $[t_{n-1}, t_n)$, and the **agent's execution value** $V_n^{(\varphi,\vartheta)}$ is the total cost (executed quantities times execution prices) over the same sub-interval. These quantities are computed as follows.

Contribution of market orders. We define the **ask-price ladder**

$$A_n^{(k)} = \widetilde{S}_n - \tfrac{1}{2}\widetilde{\varepsilon}_n + k\,\epsilon, \qquad k \in \{1, \dots, K\}. \tag{2.22}$$

Assume the agent requested to buy x units against the ask side $\widetilde{\mathbf{a}}_n = (\widetilde{\mathrm{a}}_n^{(1)}, \dots, \widetilde{\mathrm{a}}_n^{(K)})$. The **depleted liquidity** from queue $\widetilde{\mathrm{a}}_n^{(k)}$ is given by

$$d^{(k)}(\widetilde{\mathbf{a}}_n, x) = \min\big\{\, \widetilde{\mathrm{a}}_n^{(k)},\ (x - \Sigma_{k-1}(\widetilde{\mathbf{a}}_n))_+ \big\}. \tag{2.23}$$

Formula (2.23) simply breaks down x units into k amounts that do not exceed the ask queues. If we denote the **actual executed quantity** by

$$m_n(x) = \sum_{k=1}^{K} d^{(k)}(\widetilde{\mathbf{a}}_n; x), \tag{2.24}$$

then the **average price** $A_n^{(0)}(x)$ for the x units purchased is given by

$$A_n^{(0)}(x) \;=\; \begin{cases} \dfrac{\sum_{k=1}^K d^{(k)}(\widetilde{\mathbf{a}}_n, x) A_n^{(k)}}{m_n(x)}, & \text{if } m_n(x) > 0, \\ 0, & \text{if } m_n(x) = 0. \end{cases} \tag{2.25}$$

The quantity and value accruals from the MO are:

$$m_n(\varphi_n), \qquad m_n(\varphi_n) A_n^{(0)}(\varphi_n). \tag{2.26}$$

Contribution of previous limit orders. We define the **bid-price ladder** as

$$B_n^{(k)} = \widetilde{S}_n + \tfrac{1}{2}\widetilde{\varepsilon}_n - k\,\epsilon, \qquad k \in \{1,\dots,K\}. \tag{2.27}$$

Suppose the agent submitted x LOs at price $B_{n-1}^{(\ell)}$ at time t_{n-1}, $\ell \in \{1,\dots,L\}$. The **filled quantity** at time t_n is given by

$$r_n^{(\ell)}(x) = e_n^{(\ell)}\,x, \tag{2.28}$$

where $e_n^{(\ell)} \in \{0,\,0.5,\,1\}$ reflects realized activity on $[t_{n-1},t_n)$: 1 if the market traded below $B_{n-1}^{(\ell)}$, 0.5 if the lowest trade was exactly $B_{n-1}^{(\ell)}$, and 0 otherwise.

The quantity and value accruals from filled LOs are:

$$\sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)}), \qquad \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)})\,B_{n-1}^{(\ell)}. \tag{2.29}$$

Adding up all contributions. At time t_n we have

$$Q_n^{(\varphi,\vartheta)} = m_n(\varphi_n) \;+\; \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)}), \tag{2.30}$$

$$V_n^{(\varphi,\vartheta)} = m_n(\varphi_n)A_n^{(0)}(\varphi_n) \;+\; \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)})\,B_n^{(\ell)}. \tag{2.31}$$

Note that $m_0(\varphi_{-1})=0$ and $r_0^{(\ell)}(\vartheta_{-1}^{(\ell)})=0$, in line with $(\varphi_{-1},\boldsymbol{\vartheta}_{-1})=\mathbf{0}$.

Unexecuted LOs are canceled at each period end; hence no outstanding orders carry across periods. Because executions within a price level are determined exogenously, canceling an unfilled amount $(1-e_n^{(\ell)})\vartheta_{n-1}^{(\ell)}$ and submitting a new amount ϑ_n is therefore equivalent to adjusting the outstanding size by $\vartheta_n^{(\ell)} - (1-e_n^{(\ell)})\vartheta_{n-1}^{(\ell)}$. No time priority is lost, and the agent does not forgo execution "priority" under this setup.

For the last time step, we assume no LOs are submitted.

2.2.2 VWAP Policy Optimization

The **agent's total execution quantity** and **agent's average execution price** generated by the policy (φ,ϑ) are then

$$Q^{(\varphi,\vartheta)} = \sum_{n=0}^{N-1} Q_n^{(\varphi,\vartheta)}, \qquad P^{(\varphi,\vartheta)} = \begin{cases} \dfrac{\sum_{n=0}^{N-1} V_n^{(\varphi,\vartheta)}}{Q^{(\varphi,\vartheta)}}, & \text{if } Q^{(\varphi,\vartheta)} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.32}$$

VWAP is calculated from the market trades over $[0,T]$, excluding the agent's activity, since the liquidity provider does not route the agent's orders directly to the market. Let $\{(\tau_m,s_m,q_m)\}_{m=1}^{M_T}$ be the sequence of MO events over $[0,T]$, with $s_m \in \{+1,-1\}$ (buy $=+1$, sell $=-1$) and $q_m > 0$.

Define $d^{(k)}(\mathbf{b}_t, x)$ analogously to $d^{(k)}(\mathbf{a}_t, x)$, and let τ_m^- denote the immediate time before the event m. Then, the average price for each MO is given by

$$p_m = \begin{cases} \dfrac{\sum_{k=1}^{K} d^{(k)}(\mathbf{a}_{\tau_m^-}; q_m) A_{\tau_m^-}^{(k)}}{q_m}, & \text{if } s_m = +1, \\ \dfrac{\sum_{k=1}^{K} d^{(k)}(\mathbf{b}_{\tau_m^-}; q_m) B_{\tau_m^-}^{(k)}}{q_m}, & \text{if } s_m = -1. \end{cases} \tag{2.33}$$

For each sub-interval $[t_{n-1}, t_n)$, $n = 0, \dots, N$, define the set

$$M_n = \{\, m : \tau_m \in [t_{n-1}, t_n) \,\}, \tag{2.34}$$

with the convention that no trade occurred at sub-interval $[t_{-1}, t_0)$.

Aggregate trades within each interval to obtain the **market's execution quantity** Q_n^{mkt} and **market's execution value** V_n^{mkt} at time t_n:

$$Q_n^{\text{mkt}} = \sum_{m \in M_n} q_m, \qquad V_n^{\text{mkt}} = \sum_{m \in M_n} q_m \, p_m. \tag{2.35}$$

The **volume-weighted average price** (VWAP) is given by

$$\text{VWAP} = \begin{cases} \dfrac{\sum_{n=0}^{N} V_n^{\text{mkt}}}{\sum_{n=0}^{N} Q_n^{\text{mkt}}}, & \text{if } \sum_{n=0}^{N} Q_n^{\text{mkt}} > 0, \\ \text{convention (e.g., TWAP)}, & \text{otherwise.} \end{cases} \tag{2.36}$$

The **agent's terminal wealth** $W_{\text{VWAP}}^{(\varphi,\vartheta)}$ is defined as

$$W_{\text{VWAP}}^{(\varphi,\vartheta)} = Q_0\big(\text{VWAP} - \widetilde{S}_N\big) - Q^{(\varphi,\vartheta)}\big(P^{(\varphi,\vartheta)} - \widetilde{S}_N\big) - \tfrac{1}{2}\widetilde{\varepsilon}_N \left|Q^{(\varphi,\vartheta)} - Q_0\right|, \tag{2.37}$$

where we include a terminal penalty $\frac{1}{2}\widetilde{\varepsilon}_N \left|Q^{(\varphi,\vartheta)} - Q_0\right|$ to discourage over- or under-filling the mandate. This penalty approximates the cost of unwinding the residual inventory $Q^{(\varphi,\vartheta)} - Q_0$ at the terminal best quotes: selling at the bid if long (positive residual) or buying at the ask if short (negative residual). This interpretation assumes sufficient depth at the top of the book to liquidate the residual inventory.

To evaluate and rank the risk of terminal-wealth distributions $W_{\text{VWAP}}^{(\varphi,\vartheta)}$ induced by (φ, ϑ), we employ a **convex risk measure** $\rho : \mathcal{X} \to \mathbb{R}$, which is a map assigning to each real-valued r.v. a real number that satisfies *monotonicity* (a wealth distribution with greater risk has a higher risk measure), *convexity* (diversification can reduce risk), and *translation invariance* (holding risk-free cash reduces risk exactly in that cash amount). See [11] for background.

Given a convex risk measure that reflects the agent's risk preferences (e.g., Expected Shortfall for agents concerned with tail risk), the resulting optimization problem is formulated as:

$$\pi = \inf_{(\varphi,\vartheta) \in \mathcal{H}} \rho\Big(W_{\text{VWAP}}^{(\varphi,\vartheta)}\Big)\,. \tag{2.38}$$

2.3 Deep Hedging Approach

In this section, we embed the VWAP risk–hedging problem into the DH framework. We first recall the DH formulation of [3]: minimization of a convex risk measure of terminal wealth under trading frictions using a neural network policy. We then specialize this formulation to our particular setting.

2.3.1 General Formulation

We work with the same time partition and probability space $(\Omega, \mathbb{F}, \mathbb{P})$ defined in Subsection 2.2.1. Consider $d \in \mathbb{N}$ tradable assets with $\mathbb{F}$-adapted mid-price process

$$X = \{\mathbf{X}_n\}_{n=0}^{N}, \qquad \mathbf{X}_n = (X_n^{(1)}, \dots, X_n^{(d)}) \in \mathbb{R}^d_{>0}. \tag{2.39}$$

Denote the $\mathbb{F}$–adapted policy process (proposed *changes in holdings*) by

$$\delta = \{\boldsymbol{\delta}_n\}_{n=0}^{N-1}, \qquad \boldsymbol{\delta}_n = (\delta_n^{(1)}, \dots, \delta_n^{(d)}) \in \mathbb{R}^d. \tag{2.40}$$

Let $\mathcal{H}$ represent the space of all such policy processes.

Trading at t_n is subject to an $\mathcal{F}_n$–measurable, continuous function $H_n : \mathbb{R}^{d(n+1)} \to \mathbb{R}^d$ which maps the policy history $(\boldsymbol{\delta}_0, \dots, \boldsymbol{\delta}_n)$ to a traded quantity $H_n(\boldsymbol{\delta}_0, \dots, \boldsymbol{\delta}_n)$. The actual position is then given by

$$\tilde{\boldsymbol{\delta}}_n = \sum_{i=0}^{n} H_i(\boldsymbol{\delta}_0, \dots, \boldsymbol{\delta}_i), \qquad \tilde{\boldsymbol{\delta}}_{-1} = 0. \tag{2.41}$$

Profits (or losses) are calculated via the stochastic integral

$$(H \circ \delta \bullet X)_N = \sum_{n=0}^{N-1} \tilde{\boldsymbol{\delta}}_n \cdot (\mathbf{X}_{n+1} - \mathbf{X}_n). \tag{2.42}$$

Transaction costs at each node are modeled by $\mathcal{F}_n$–measurable continuous functions $c_n : \mathbb{R}^d \to \mathbb{R}$ that assign to the vector of traded quantities $\Delta\tilde{\boldsymbol{\delta}}_n$ a per-period cost $c_n\big(\Delta\tilde{\boldsymbol{\delta}}_n\big)$. The total cost is then

$$C_N(H \circ \delta) = \sum_{n=0}^{N-1} c_n\big(\Delta\tilde{\boldsymbol{\delta}}_n\big), \qquad \Delta\tilde{\boldsymbol{\delta}}_n = \tilde{\boldsymbol{\delta}}_n - \tilde{\boldsymbol{\delta}}_{n-1} = H_n(\boldsymbol{\delta}_0, \dots, \boldsymbol{\delta}_n). \tag{2.43}$$

Assuming that the agent faces a $\mathcal{F}_N$-measurable liability Z at T, the terminal wealth is given by

$$W^\delta(Z) = -Z \; + \; (H \circ \delta \bullet X)_N \; - \; C_N(H \circ \delta). \tag{2.44}$$

The optimization objective is

$$\pi(Z) = \inf_{\delta \in \mathcal{H}} \; \rho\big(W^\delta(Z)\big)\,, \tag{2.45}$$

where ρ was introduced in Subsection 2.2.2.

2.3.2 Solution via Neural Networks

Following [3], we parameterize the policy (2.40) using a time-indexed family of neural networks (defined below). Since $\mathcal{F}_n = \sigma(\mathbf{I}_0, \dots, \mathbf{I}_n)$ and $\boldsymbol{\delta}_n$ is $\mathcal{F}_n$-measurable, there exists a measurable map $f_n : \mathbb{R}^{r(n+1)} \to \mathbb{R}^d$ such that $\boldsymbol{\delta}_n = f_n(\mathbf{I}_0, \dots, \mathbf{I}_n)$ for $n = 0, \dots, N-1$. Solving (2.45) by optimizing over all such families $\{f_n(\cdot)\}$ is an infinite-dimensional problem. Instead, we restrict our attention to a parametric class $\{f_n(\theta_n; \cdot)\}$ and approximate a minimizer of (2.45) by solving the finite-dimensional problem in the parameters $\boldsymbol{\theta} = (\theta_0, \dots, \theta_{N-1})$.

Definition 2.4 (Neural network)**.** Let $L, H_0, H_1, \dots, H_L, H_{L+1} \in \mathbb{N}$ be given. For each $\ell = 1, \dots, L+1$, take weights $A^{(\ell)} \in \mathbb{R}^{H_\ell \times H_{\ell-1}}$ and biases $b^{(\ell)} \in \mathbb{R}^{H_\ell}$. With $h^{(0)} = x \in \mathbb{R}^{H_0}$, define for $\ell = 1, \dots, L$ the hidden layers

$$h^{(\ell)} = \sigma\big(A^{(\ell)} h^{(\ell-1)} + b^{(\ell)}\big) \in \mathbb{R}^{H_\ell}, \tag{2.46}$$

where the continuous, non-polynomial map $\sigma : \mathbb{R} \to \mathbb{R}$ is applied entry-wise.

The function $f(\theta; \cdot) : \mathbb{R}^{H_0} \to \mathbb{R}^{H_{L+1}}$ given by

$$f(\theta; x) = A^{(L+1)} h^{(L)} + b^{(L+1)} \tag{2.47}$$

is a **neural network** with parameters $\theta = \{(A^{(\ell)}, b^{(\ell)})\}_{\ell=1}^{L+1}$, input dimension H_0, output dimension H_{L+1}, and L hidden layers of widths $H_1, \dots, H_L$. We refer to the number of hidden layers and corresponding layer widths as **architecture** of the network.

We build a nondecreasing family of architectures, along fixed input/output dimensions. For this, let $\{L^{(M)}\}_{M \in \mathbb{N}}$ be a nondecreasing depth sequence. For each M, consider the layer widths $H_0^{(M)}, H_1^{(M)}, \dots, H_{L(M)}^{(M)}, H_{L(M)+1}^{(M)}$ with boundary conditions

$$H_0^{(M)} = d_{\text{in}} \in \mathbb{N}, \qquad H_{L^{(M)}+1}^{(M)} = d_{\text{out}} \in \mathbb{N}, \tag{2.48}$$

and assume that for every $\ell \in \{1, \dots, L(M)\}$, the sequence $\{H_\ell^{(M)}\}_{M \in \mathbb{N}}$ is nondecreasing in M.

For a fixed architecture (i.e., fixed depth $L^{(M)}$ and widths $H_1^{(M)}, \dots, H_{L(M)}^{(M)}$), define the **parameter space** with input dimension d_{in} and output dimension d_{out} as

$$\Theta_M(d_{\text{in}}, d_{\text{out}}) = \Big\{\{(A^{(\ell)}, b^{(\ell)})\}_{\ell=1}^{L^{(M)}+1} : A^{(\ell)} \in \mathbb{R}^{H_\ell^{(M)} \times H_{\ell-1}^{(M)}}, b^{(\ell)} \in \mathbb{R}^{H_\ell^{(M)}}\Big\}. \tag{2.49}$$

For time step $n \in \{0, \dots, N-1\}$, take $d_{\text{in}} = r(n+1) + d$ and $d_{\text{out}} = d$, and define the **total parameter space** as the product

$$\Theta_M = \prod_{n=0}^{N-1} \Theta_M\big(r(n+1) + d, d\big). \tag{2.50}$$

The next result shows that, for sufficiently large architectures, a recursive policy network approximates the optimal policy that solves (2.45).

Proposition 2.5 (Buehler et al. [3]). *Let $\boldsymbol{\theta} = (\theta_0, \ldots, \theta_{N-1}) \in \Theta_M$, and take a sequence of neural networks*

$$\{f_n(\theta_n; \cdot) : \mathbb{R}^{r(n+1)+d} \to \mathbb{R}^d\}_{n=0}^{N-1}. \tag{2.51}$$

Define the policy $\delta^{\boldsymbol{\theta}} = \{\boldsymbol{\delta}_n\}_{n=0}^{N-1}$ recursively by

$$\boldsymbol{\delta}_{-1} = \mathbf{0}; \qquad \boldsymbol{\delta}_n = f_n\big(\theta_n; \mathbf{I}_0, \ldots, \mathbf{I}_n, \boldsymbol{\delta}_{n-1}\big), \quad n = 0, \ldots, N-1. \tag{2.52}$$

Consider the optimization objective

$$\pi_M(Z) = \inf_{\boldsymbol{\theta} \in \Theta_M} \rho\Big(W^{\delta^{\boldsymbol{\theta}}}(Z)\Big). \tag{2.53}$$

Then, for any $Z \in \mathcal{X}$,

$$\lim_{M \to \infty} \pi_M(Z) = \pi(Z). \tag{2.54}$$

2.3.3 Application to VWAP Policy Optimization

By modeling the components in (2.44), we express $W_{\mathrm{VWAP}}^{(\varphi,\vartheta)}$ as the sum of a liability, transaction costs, and trading gains. This justifies the use of neural networks to approximate the optimal policy through Proposition 2.5.

First, we show that if transaction costs decompose asset-wise via per-asset execution prices, then the after-cost trading gains equal the mark-to-market of separate long and short books in each asset, benchmarked at terminal mid-prices.

Proposition 2.6. *Let $\boldsymbol{y}_n \in \mathbb{R}^d$ denote the actual asset position at time t_n, and suppose there exist functions $p_n^{(i)} : \mathbb{R} \to \mathbb{R}_{\geq 0}$ such that*

$$c_n\big(\Delta \boldsymbol{y}_n\big) = \sum_{i=1}^{d} \Delta y_n^{(i)} \big(p_n^{(i)}(\Delta y_n^{(i)}) - X_n^{(i)}\big). \tag{2.55}$$

Then,

$$(H \circ \delta \bullet X)_N - C_N(H \circ \delta) = \sum_{i=1}^{d} Q^{i,a}\big(P^{i,a} - X_N^{(i)}\big) - \sum_{i=1}^{d} Q^{i,b}\big(P^{i,b} - X_N^{(i)}\big), \tag{2.56}$$

where $Q^{i,a}$ and $Q^{i,b}$ are the cumulative short and long quantities in asset i, and $P^{i,a}$ and $P^{i,b}$ are the corresponding average sale and buy prices.

Proof. Fix $i \in \{1, \ldots, d\}$ and abbreviate $p_n^{(i)}(\Delta y_n^{(i)})$ by $p_n^{(i)}$. Using the discrete integration-by-parts identity

$$\sum_{n=0}^{N-1} y_n^{(i)} \, \Delta X_{n+1}^{(i)} = y_{N-1}^{(i)} X_N^{(i)} - \sum_{n=0}^{N-1} \Delta y_n^{(i)} X_n^{(i)} \tag{2.57}$$

with $y_{-1}^{(i)} = 0$, we obtain

$$(H \circ \delta \bullet X)_N - C_N(H \circ \delta) = \sum_{i=1}^{d} \Big(y_{N-1}^{(i)} X_N^{(i)} - \sum_{n=0}^{N-1} \Delta y_n^{(i)} p_n^{(i)}\Big). \tag{2.58}$$

Write $\Delta y_n^{i,b} := (\Delta y_n^{(i)})_+$ and $\Delta y_n^{i,a} := (-\Delta y_n^{(i)})_+$. Then,

$$y_{N-1}^{(i)} = \sum_{n=0}^{N-1} \Delta y_n^{i,b} - \sum_{n=0}^{N-1} \Delta y_n^{i,a}, \tag{2.59}$$

$$\sum_{n=0}^{N-1} \Delta y_n^{(i)} p_n^{(i)} = \sum_{n=0}^{N-1} \Delta y_n^{i,b} p_n^{(i)} - \sum_{n=0}^{N-1} \Delta y_n^{i,a} p_n^{(i)}. \tag{2.60}$$

Replacing (2.59) and (2.60) in the i-th summand in (2.58) and collecting terms yields

$$Q^{i,a}\big(P^{i,a} - X_N^{(i)}\big) - Q^{i,b}\big(P^{i,b} - X_N^{(i)}\big), \tag{2.61}$$

where

$$Q^{i,a} = \sum_{n=0}^{N-1} \Delta y_n^{i,a}, \qquad P^{i,a} = \frac{\sum_{n=0}^{N-1} \Delta y_n^{i,a} p_n^{(i)}}{Q^{i,a}}, \tag{2.62}$$

$$Q^{i,b} = \sum_{n=0}^{N-1} \Delta y_n^{i,b}, \qquad P^{i,b} = \frac{\sum_{n=0}^{N-1} \Delta y_n^{i,b} p_n^{(i)}}{Q^{i,b}}, \tag{2.63}$$

with the conventions $P^{i,a} = 0$ if $Q^{i,a} = 0$ and $P^{i,b} = 0$ if $Q^{i,b} = 0$. Summing over i gives (2.56). □

We now choose a $\mathcal{F}_N$-measurable r.v. Z, $\mathbb{F}$-adapted processes X and δ, and $\mathcal{F}_n$-measurable, continuous functions H_n and c_n to model (2.37) in terms of (2.44). To this end, set

$$X_n^{(\ell)} = \widetilde{S}_n, \quad \ell = 1, \dots, L+1, \tag{2.64}$$

$$\boldsymbol{\delta}_n = \big(\varphi_n, \vartheta_n^{(1)}, \dots, \vartheta_n^{(L)}\big), \tag{2.65}$$

$$H_n(\boldsymbol{\delta}_0, \dots, \boldsymbol{\delta}_n) = \Big(m_n(\varphi_n),\ r_n^{(1)}(\vartheta_{n-1}^{(1)}),\ \dots,\ r_n^{(L)}(\vartheta_{n-1}^{(L)})\Big), \tag{2.66}$$

$$c_n(H_n) = m_n(\varphi_n)\Big(A_n^{(0)}(\varphi_n) - \widetilde{S}_n\Big) + \sum_{\ell=1}^{L} r_n^{(\ell)}(\vartheta_{n-1}^{(\ell)})\big(B_{n-1}^{(\ell)} - \widetilde{S}_n\big), \tag{2.67}$$

$$Z = Q_0\big(\widetilde{S}_N - \mathrm{VWAP}\big) + \tfrac{1}{2}\widetilde{\varepsilon}_N \left|Q^{(\varphi,\vartheta)} - Q_0\right|, \tag{2.68}$$

where $Q^{(\varphi,\vartheta)}$ is given in (2.32). Here, $X_n^{(\ell)}$ runs from 0 through N; $\boldsymbol{\delta}_n$, H_n, and c_n stop at $N-1$.

Observe that c_n already has the form of (2.55). By Proposition 2.6,

$$(H \circ \delta \bullet X)_N - C_N(H \circ \delta) = -Q^{(\varphi,\vartheta)}(P^{(\varphi,\vartheta)} - \widetilde{S}_N), \tag{2.69}$$

where $P^{(\varphi,\vartheta)}$ was defined in (2.32). Subtracting Z yields (2.37).

Remark 2.7. While DH was originally developed for trading multiple distinct assets, it also supports an alternative application for trading on a *single* underlying with *several mandates*: the $L+1$ components act as separate execution books (one active MO book and L passive LO books at successive bid levels). The active book buys at prevailing ask levels; the passive books attempt to fill at the best bid level, second-best bid level, etc. The decomposition (2.56) then aggregates these books into one single result.

2.3.4 Market Features

At time t_n, the relevant market information includes:

- The LOB state: $\widetilde{\mathbf{L}}_n = (\widetilde{S}, \widetilde{\varepsilon}, \widetilde{\mathbf{a}}, \widetilde{\mathbf{b}})_n$.
- The realized LO fill coefficients from the previous interval at the initial bid levels: $e_n^{(1)}, \dots, e_n^{(L)}$.
- The cumulative market's execution quantity and value: $\sum_{i=0}^{n} Q_i^{\mathrm{mkt}}$, $\sum_{i=0}^{n} V_i^{\mathrm{mkt}}$.

The number of LO relative prices L is computed via a data–driven rule on the training set. For trajectory $j \in \{1, \dots, J\}$ on the training set and index $n \in \{1, \dots, N-1\}$, define the **maximal reached level** in $[t_{n-1}, t_n)$ by

$$\ell_{j,n} = \sup_{k \in \{1,\dots,K\}} \big\{k : e_{j,n}^{(k)} > 0\big\}, \tag{2.70}$$

where $e_{j,n}^{(k)} = e_n^{(k)}$ on the j-th trajectory. We set L to the empirical 99% quantile of $\{\ell_{j,n}\}$ over all (j, n).

To reduce input dimension, we truncate the observable depth K by passing only the first L queues per side, thereby retaining the transacted region in the vast majority of intervals while avoiding unnecessary features from rarely used deeper levels.

The observable information at time t_n is then gathered into the feature vector

$$\mathbf{I}_n = \Big(\widetilde{S}_n, \widetilde{\varepsilon}_n, \widetilde{\mathrm{a}}_n^{(1)}, \dots, \widetilde{\mathrm{a}}_n^{(L)}, \widetilde{\mathrm{b}}_n^{(1)}, \dots, \widetilde{\mathrm{b}}_n^{(L)}, e_n^{(1)}, \dots, e_n^{(L)}, \sum_{i=0}^{n} Q_i^{\mathrm{mkt}}, \sum_{i=0}^{n} V_i^{\mathrm{mkt}}\Big). \tag{2.71}$$

Remark 2.8. Under the Santa Fe model, the time-discrete LOB state $\widetilde{\mathrm{L}}$ is a $(\mathbb{F}, \mathbb{P})$-Markov process. Therefore, as pointed out in [3], we may simplify the recursive policy (2.52) as follows:

$$(\varphi_n, \vartheta_n^{(1)}, \dots, \vartheta_n^{(L)}) = f_n\big(\theta_n; \mathbf{I}_n, \varphi_{n-1}, \vartheta_{n-1}^{(1)}, \dots, \vartheta_{n-1}^{(L)}\big), \quad n = 0, \dots, N-1, \tag{2.72}$$

with $\big(\varphi_{-1}, \vartheta_{-1}^{(1)}, \dots, \vartheta_{-1}^{(L)}\big) = \mathbf{0}$. Consequently, all sub-networks have identical input dimension, allowing the total parameter space to be expressed more compactly as

$$\widetilde{\Theta}_M = \prod_{n=0}^{N-1} \Theta_M\big(r + d, d\big). \tag{2.73}$$

To compute wealth samples, we use **Algorithm 2**: given simulated market features $\{(\mathbf{I}_0^{(j)}, \mathbf{I}_1^{(j)}, \dots, \mathbf{I}_N^{(j)})\}_{j=1}^{J}$ and network parameters $\boldsymbol{\theta} = (\theta_0, \dots, \theta_{N-1}) \in \widetilde{\Theta}_M$, the algorithm loops over scenarios $j \in \{1, \dots, J\}$. For each j, it initializes the previous policy to zero and sets cumulative trading gains G and costs C to zero. Then, for each time step $n \in \{0, \dots, N-1\}$, it computes the policy via (2.72), updates trading gains and trading costs, and rolls the policy forward. After the time loop, it computes the liability Z (which includes the VWAP benchmark and terminal penalty), and records the wealth sample W. A collection of J wealth samples is returned for use in the chosen risk objective.

Algorithm 2: Compute wealth samples

```
inputs: simulated market features $\{(\mathbf{I}_0^{(j)}, \mathbf{I}_1^{(j)}, \dots, \mathbf{I}_N^{(j)})\}_{j=1}^{J}$; neural network parameters $\boldsymbol{\theta} = (\theta_0, \dots, \theta_{N-1})$; target quantity $Q_0$
for $j = 1$ to $J$ do
    initialize: $(\varphi_{-1}, \vartheta_{-1}^{(1)}, \dots, \vartheta_{-1}^{(L)}) \leftarrow \mathbf{0}$, $Q \leftarrow 0$, $G \leftarrow 0$, $C \leftarrow 0$
    for $n = 0$ to $N - 1$ do
        Compute policy: $(\varphi, \vartheta^{(1)}, \dots, \vartheta^{(L)}) \leftarrow f_n(\theta_n; \mathbf{I}_n^{(j)}, \varphi_{-1}, \vartheta_{-1}^{(1)}, \dots, \vartheta_{-1}^{(L)})$
        Update inventory: $Q \leftarrow Q + Q_n^{(\varphi,\vartheta),j}$
        Update profits: $G \leftarrow G + Q(\widetilde{S}_{n+1}^{(j)} - \widetilde{S}_n^{(j)})$
        Update costs: $C \leftarrow C + V_n^{(\varphi,\vartheta),j} - Q_n^{(\varphi,\vartheta),j}\widetilde{S}_n^{(j)}$
        Update previous policy: $(\varphi_{-1}, \vartheta_{-1}^{(1)}, \dots, \vartheta_{-1}^{(L)}) \leftarrow (\varphi, \vartheta^{(1)}, \dots, \vartheta^{(L)})$
    end for
    Compute VWAP: $\mathrm{VWAP} \leftarrow \sum_{i=0}^{N} V_i^{\mathrm{mkt},j} / \sum_{i=0}^{N} Q_i^{\mathrm{mkt},j}$
    Compute liability: $Z \leftarrow Q_0(\widetilde{S}_N^{(j)} - \mathrm{VWAP}) - \frac{1}{2}\widetilde{\varepsilon}_N^{(j)}|Q - Q_0|$
    Compute wealth sample: $W^{(j)}(\boldsymbol{\theta}) \leftarrow -Z + G - C$
end for
output: $\{(W^{(1)}(\boldsymbol{\theta}), \dots, W^{(J)}(\boldsymbol{\theta})\}$
```

2.4 Loss Function Minimization

In this section, we outline the procedure to optimize the policy network parameters when the agent uses Expected Shortfall as risk measure. Ideally, we would use SGD to find the optimal parameters; however, mini-batch gradients for loss functions that are convex risk measures are biased estimators of their full-batch version. As a work-around, we extend the search space by one dimension and perform mini-batch optimization of an equivalent representation of Expected Shortfall. For this, we use the Rockafellar-Uryasev objective in [12].

We fix a shared architecture for every neural network; that is, we fix the same depth $L^{(M)}$ and widths $H_1^{(M)}, \dots, H_{L(M)}^{(M)}$ for all $g_n(\theta_n; \cdot)$, $n \in \{1, \dots, N-1\}$.

2.4.1 Rockafellar–Uryasev Objective

Take parameters $\boldsymbol{\theta} \in \widetilde{\Theta}_M$, and denote by $\{W^{(1)}(\boldsymbol{\theta}), \dots, W^{(J)}(\boldsymbol{\theta})\}$ the wealth samples computed via **Algorithm 2**. Define the losses $L^{(j)}(\boldsymbol{\theta}) = -W^{(j)}(\boldsymbol{\theta})$, $j \in \{1, \dots, J\}$, and let $\ell^{(1)}(\boldsymbol{\theta}) \le \cdots \le \ell^{(J)}(\boldsymbol{\theta})$ be the sorted loss statistic.

Let $\alpha \in (0,1)$ (confidence level) and $\tau \in \mathbb{R}$ (threshold), and define $k = \lceil \alpha J \rceil$, $r = k - \alpha J$. The (sample) **Expected Shortfall** (ES) is given by

$$\widehat{\mathrm{ES}}_\alpha(\boldsymbol{\theta}) = \frac{1}{(1-\alpha)\,J}\left(\sum_{j=k+1}^{J} \ell^{(j)}(\boldsymbol{\theta}) + r\,\ell^{(k)}(\boldsymbol{\theta})\right). \tag{2.74}$$

The (sample) **Rockafellar–Uryasev** objective is

$$\widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta}) = \tau + \frac{1}{(1-\alpha)\,J}\sum_{j=1}^{J}\left(L^{(j)}(\boldsymbol{\theta}) - \tau\right)_+. \tag{2.75}$$

The following result links both metrics:

Theorem 2.9 (Rockafellar and Uryasev [12]). *For any $\alpha \in (0,1)$,*

$$\widehat{\mathrm{ES}}_\alpha(\boldsymbol{\theta}) = \min_{\tau \in \mathbb{R}} \widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta}), \tag{2.76}$$

with minimizer

$$\tau^*(\boldsymbol{\theta}) \in \arg\min_{\tau \in \mathbb{R}} \widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta}) = \begin{cases} \ell^{(k)}(\boldsymbol{\theta}), & \text{if } \alpha J \notin \mathbb{N}, \\ [\ell^{(k)}(\boldsymbol{\theta}), \ell^{(k+1)}(\boldsymbol{\theta})], & \text{otherwise .} \end{cases} \tag{2.77}$$

By Theorem 2.9 and the fact that the minimization of $\widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta})$ with respect to $(\tau, \boldsymbol{\theta}) \in \mathbb{R} \times \Theta_M$ can be carried out by first minimizing over $\tau \in \mathbb{R}$ for fixed $\boldsymbol{\theta}$ and then minimizing the result over $\boldsymbol{\theta} \in \Theta_M$ (see [12]), we have

$$\min_{\boldsymbol{\theta} \in \Theta_M} \widehat{\mathrm{ES}}_\alpha(\boldsymbol{\theta}) = \min_{(\tau,\boldsymbol{\theta}) \in \mathbb{R} \times \Theta_M} \widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta}), \tag{2.78}$$

where a pair $(\tau^*, \boldsymbol{\theta}^*)$ minimizes the right-hand-side of (2.78) if and only if $\boldsymbol{\theta}^*$ minimizes the left-hand-side of (2.78) and $\tau^* \in \arg\min_{\tau \in \mathbb{R}} \widehat{\mathrm{RU}}_{\alpha,\tau}(\boldsymbol{\theta}^*)$.

Equation (2.78) allows us to minimize ES via the equivalent RU objective, the advantage being that the latter is suitable for mini-batch training. This requires the map $\boldsymbol{\theta} \mapsto W^{(j)}(\boldsymbol{\theta})$ to be (piecewise) differentiable, which we show next.

Proposition 2.10. *Assume the nonlinearity σ is (piecewise) C^1. Then, the map $\boldsymbol{\theta} \mapsto W^{(j)}(\boldsymbol{\theta})$ is (piecewise) differentiable.*

Proof. Because φ_n and $\vartheta_n^{(\ell)}$ are neural network outputs, each is a composition of affine maps and σ (element-wise). Since σ is (piecewise) C^1, it follows that $\nabla_{\boldsymbol{\theta}} \varphi_n$ and $\nabla_{\boldsymbol{\theta}} \vartheta_n^{(\ell)}$ are valid (sub)gradients, justifying their use in the next formulas:

$$\nabla_{\boldsymbol{\theta}} Q^{(j)}(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} \left(\mathbf{1}_{\{\varphi_n < \Sigma_K(\widetilde{\mathbf{a}}_n)\}} \nabla_{\boldsymbol{\theta}} \varphi_n + \sum_{\ell=1}^{L} e_n^{(\ell)} \nabla_{\boldsymbol{\theta}} \vartheta_{n-1}^{(\ell)} \right), \tag{2.79}$$

$$\nabla_{\boldsymbol{\theta}} V^{(j)}(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} \left(\sum_{k=1}^{K} \mathbf{1}_{\{\Sigma_{k-1}(\widetilde{\mathbf{a}}_n) < \varphi_n < \Sigma_k(\widetilde{\mathbf{a}}_n)\}} A_n^{(k)} \nabla_{\boldsymbol{\theta}} \varphi_n + \sum_{\ell=1}^{L} e_n^{(\ell)} B_{n-1}^{(\ell)} \nabla_{\boldsymbol{\theta}} \vartheta_{n-1}^{(\ell)} \right). \tag{2.80}$$

From (2.37) and the fact that $Q^{(\varphi,\vartheta)} P^{(\varphi,\vartheta)} = V^{(\varphi,\vartheta)}$, we have

$$\nabla_{\boldsymbol{\theta}} W^{(j)}(\boldsymbol{\theta}) = \left(\widetilde{S}_N^{(j)} - \frac{\widetilde{\varepsilon}_N}{2} \mathrm{sign}(Q^{(j)}(\boldsymbol{\theta}) - Q_0) \right) \nabla_{\boldsymbol{\theta}} Q^{(j)}(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} V^{(j)}(\boldsymbol{\theta}). \tag{2.81}$$

□

2.4.2 Parameters Update

While Proposition 2.10 justifies the use of gradient-based methods for optimizing the network parameters, it does not ensure convergence to a global minimum of (2.53). To reduce the likelihood of becoming trapped in local minima, one typically employs SGD. For SGD to be effective, however, the mini-batch (sub)gradients must be unbiased. We now analyze this condition more closely.

Let $B \subseteq \{1, \dots, J\}$ be a mini-batch sampled uniformly (with or without replacement). We define the **RU mini-batch (sub)gradients** by

$$g^B_{\boldsymbol{\theta}}(\tau, \boldsymbol{\theta}) = \frac{1}{(1-\alpha)\,|B|} \sum_{b \in B} \mathbf{1}\{L^{(b)}(\boldsymbol{\theta}) > \tau\}\, \nabla_{\boldsymbol{\theta}} L^{(b)}(\boldsymbol{\theta}), \tag{2.82}$$

$$g^B_{\tau}(\tau, \boldsymbol{\theta}) = 1 - \frac{1}{(1-\alpha)\,|B|} \sum_{b \in B} \mathbf{1}\{L^{(b)}(\boldsymbol{\theta}) > \tau\}, \tag{2.83}$$

where $\nabla_{\boldsymbol{\theta}} L^{(b)}(\boldsymbol{\theta}) = -\nabla_{\boldsymbol{\theta}} W^{(b)}(\boldsymbol{\theta})$ by definition. Note that in the particular case where B is the whole training set, we recover the **RU full-batch (sub)gradients**, denoted by $g^J_{\boldsymbol{\theta}}(\tau, \boldsymbol{\theta})$ and $g^J_{\tau}(\tau, \boldsymbol{\theta})$.

Conditioning on the dataset, a standard calculation yields

$$\mathbb{E}\big[g^B_{\boldsymbol{\theta}}(\tau, \boldsymbol{\theta})\big] = g^J_{\boldsymbol{\theta}}(\tau, \boldsymbol{\theta}), \qquad \mathbb{E}\big[g^B_{\tau}(\tau, \boldsymbol{\theta})\big] = g^J_{\tau}(\tau, \boldsymbol{\theta}), \tag{2.84}$$

which means that SGD on the RU objective (jointly in $(\tau, \boldsymbol{\theta})$) is unbiased and converges under the usual conditions.

Remark 2.11. In practice, at each epoch we adopt a two-step procedure motivated by (2.78). First, for a given $\boldsymbol{\theta}$, we compute $\tau^*(\boldsymbol{\theta})$ as characterized in (2.77). Second, we fix this value of τ and perform a SGD step on $\boldsymbol{\theta}$ using the mini-batch (sub)gradient in (2.82).

2.5 Neural Network Architecture

In this section, we specify the policy network architecture used in our experiments.

In the simplified recursion (2.72), each time–t_n sub-network maps the current information and the previous action to the next action. Concretely, each sub-network receives $3L + 5$ inputs ($2L + 4$ market features and $L + 1$ previous policy actions) and produces $L + 1$ current policy actions.

Network map. For each t_n, $n \in \{0, \dots, N-1\}$, we employ a feedforward network with two hidden layers of width 16:

$$H_0 = 3L + 5, \qquad H_\ell = 16 \;\; (\ell = 1, 2), \qquad H_3 = L + 1. \tag{2.85}$$

Taking weights $A^{(\ell)}_n \in \mathbb{R}^{H_\ell \times H_{\ell-1}}$ and biases $b^{(\ell)}_n \in \mathbb{R}^{H_\ell}$, $\ell \in \{1, 2, 3\}$, the input layer is

$$h^{(0)}_n = (\mathbf{I}_n, \varphi_{n-1}, \vartheta^{(1)}_{n-1}, \dots, \vartheta^{(L)}_{n-1}), \tag{2.86}$$

the hidden layers are

$$h^{(\ell)}_n = \tanh\big(A^{(\ell)}_n h^{(\ell-1)}_n + b^{(\ell)}_n\big), \qquad \ell = 1, 2, \tag{2.87}$$

and the output layer is

$$h^{(3)}_n = \mathrm{ReLU}\big(A^{(3)}_n h^{(2)}_n + b^{(3)}_n\big) = (\varphi_n, \vartheta^{(1)}_n, \dots, \vartheta^{(L)}_n). \tag{2.88}$$

The tanh nonlinearity supplies smooth saturating hidden features, while the output ReLU enforces the constraint of nonnegative order sizes.

Time discretization. We fix $N = 128$ trading nodes and adopt an equidistant grid on $[0, T]$, $t_n = \frac{n}{N}T,\ n \in \{0, \dots, N\}$, so that the agent rebalances at uniform intervals.

Training protocol. The training set consists of $J = 250{,}000$ simulated trajectories generated under the Santa Fe model (Subsection 2.1.3). We use early stopping with patience 10 (budget 1,000 epochs) and a halve-on-plateau learning-rate schedule (initial rate 10^{-1}, minimum 2^{-10}). Optimization is performed with a Adam (see Remark 2.11) with mini-batches of size $B = 25{,}000$. The network parameters are initialized at random.

A schematic of one sub-network and of the full unrolled policy appears below.

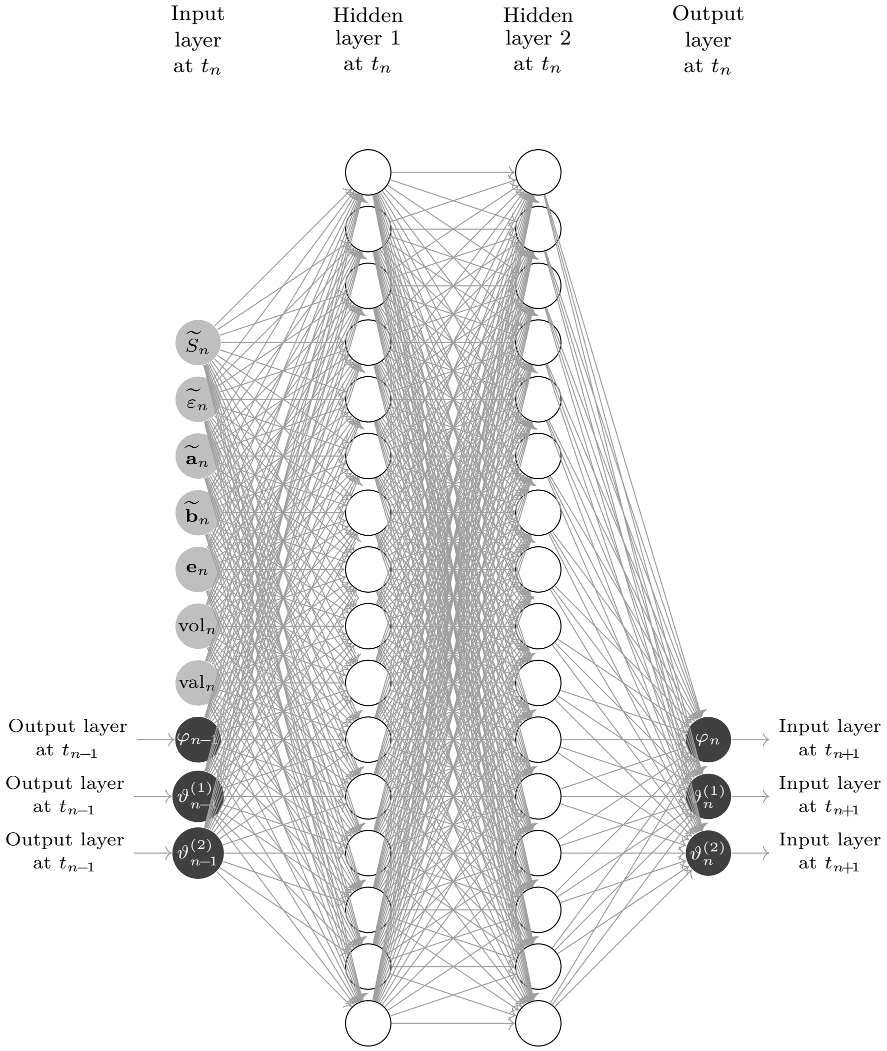

Figure 2.2: Single sub-network: ten input neurons (including three feedback inputs from the previous time step), two hidden layers of 16 neurons (`tanh`), and three outputs (`ReLU`) that are fed back at the next time step. For visual clarity, market feature vectors ($\widetilde{\mathbf{a}}_n$, $\widetilde{\mathbf{b}}_n$, and $\mathbf{e}_n$) are shown as single neurons.

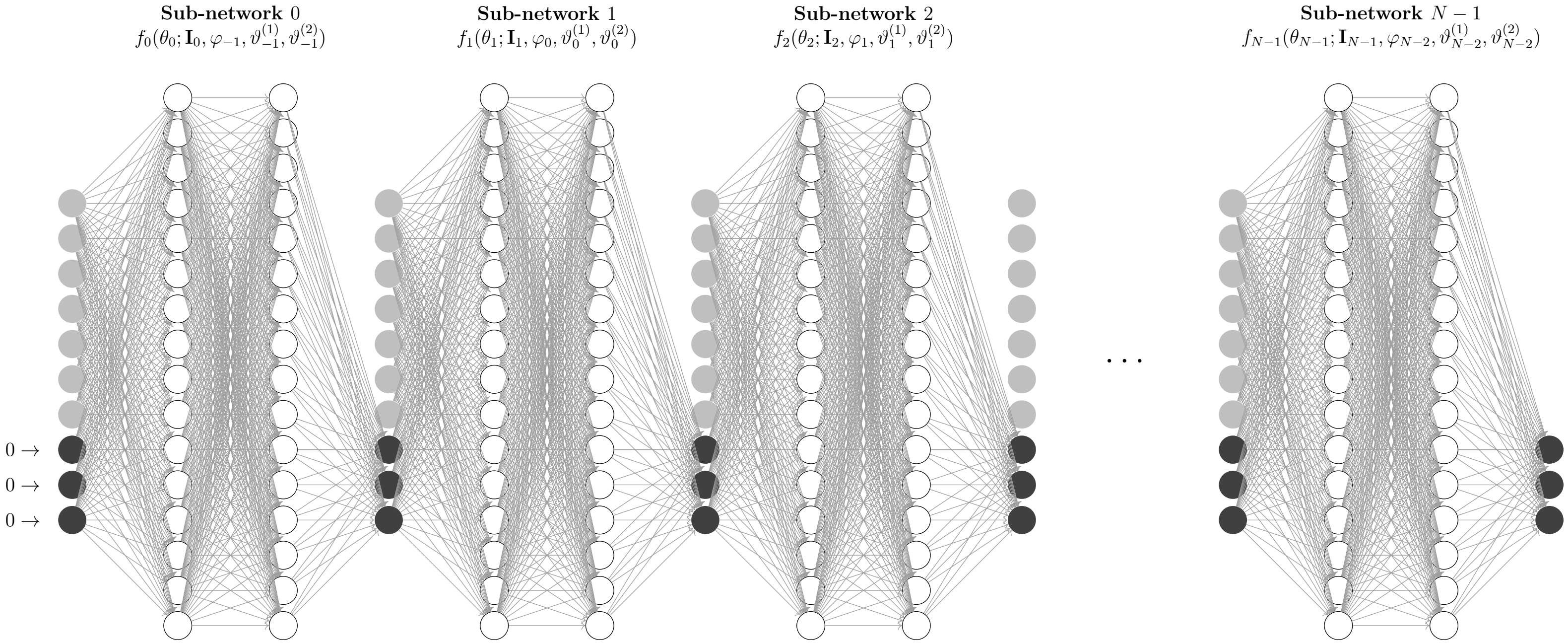
Sub-network 0
$f_0(\theta_0; \mathbf{I}_0, \varphi_{-1}, \vartheta^{(1)}_{-1}, \vartheta^{(2)}_{-1})$
Sub-network 1
$f_1(\theta_1; \mathbf{I}_1, \varphi_0, \vartheta^{(1)}_0, \vartheta^{(2)}_0)$
Sub-network 2
$f_2(\theta_2; \mathbf{I}_2, \varphi_1, \vartheta^{(1)}_1, \vartheta^{(2)}_1)$
Sub-network $N-1$
$f_{N-1}(\theta_{N-1}; \mathbf{I}_{N-1}, \varphi_{N-2}, \vartheta^{(1)}_{N-2}, \vartheta^{(2)}_{N-2})$
$0 \rightarrow$
$0 \rightarrow$
$0 \rightarrow$
$\cdots$

Figure 2.3: System with N sub-networks. Each sub-network ingests ten inputs (seven gray market features and three black previous-policy features), has two hidden layers of 16 neurons, and outputs three non-negative actions that feed the next time step. The initial black inputs are zeros (flat start).

Chapter 3

Empirical Results

In this chapter, we calibrate the Santa Fe model parameters to intraday equity LOB data. We then train the agent across distinct market environments and examine the learned trading policies, focusing on terminal wealth distribution, hedge relative to the target quantity, and average order allocations.

3.1 Estimated Parameters and Training Data

The numerical experiments are based on high-frequency data from LOBSTER (Limit Order Book Reconstruction System, [10]), which provides detailed records of submissions, cancellations, and executions, including timestamps, prices, sizes, and buy/sell indicators. For our simulations, we selected four stocks:

Name	Ticker	Start date	End date	Start time	End time
Cisco	CSCO	2015-01-01	2015-03-31	09:30	16:00
Intel	INTC	2015-01-01	2015-03-31	09:30	16:00
Priceline	PCLN	2015-01-01	2015-03-31	09:30	16:00
Tesla	TSLA	2015-01-01	2015-03-31	09:30	16:00

Table 3.1: Dataset description: ticker symbols, sample period, and daily trading window.

To calibrate the Santa Fe model, we estimate order-flow intensities and volume distribution parameters for each stock. Order volumes are modeled as log-normal.

We restrict the analysis to 10:00–15:30 to avoid open and close noise effects (e.g., liquidity imbalances, wide spreads). This yields an effective horizon of $T = 19{,}800$ seconds (5.5 trading hours per day). With $N = 128$ equally spaced sub-intervals, decision times are separated by approximately 2.5 minutes. Auction periods and hidden liquidity are excluded from the analysis.

To calibrate the parameters, we use 61 business days per stock over the period reported in **Table 3.1**.

Calibrated values are summarized in the table and figures below. The detailed estimation procedure, including LO arrival and cancellation rates by relative price, is described in Appendix A.

Parameter		CSCO	INTC	PCLN	TSLA
Tick	ϵ	0.01	0.01	0.01	0.01
LOB depth	K	10	10	295	75
Relative price levels	L	6	5	78	18
LOB bounds	c^{∞}	7,764	4,377	178	6,494
MO intensity	2γ	0.6067	0.8508	0.0936	0.2730
LO total intensity	2λ	9.7078	13.2461	2.2086	4.6423
CXL total intensity (avg.)	$\rho^{\mathrm{a}}+\rho^{\mathrm{b}}$	9.5266	12.6604	2.1232	4.4019
Total event intensity (avg.)	Λ	19.8411	26.7573	4.4254	9.3172
MO log-volume location	μ_{M}	5.6911	5.2753	3.6789	3.9220
MO log-volume scale	σ_{M}	1.0924	0.9487	1.0477	1.3312
LO log-volume location	μ_{L}	5.0833	4.8582	3.1401	4.3454
LO log-volume scale	σ_{L}	1.1052	1.0253	1.4952	0.5664
CXL log-volume location	μ_{C}	5.5645	5.2079	3.6754	4.3453
CXL log-volume scale	σ_{C}	1.3179	1.0834	1.0271	0.5161

Table 3.2: Santa Fe model parameter estimates for all stocks.

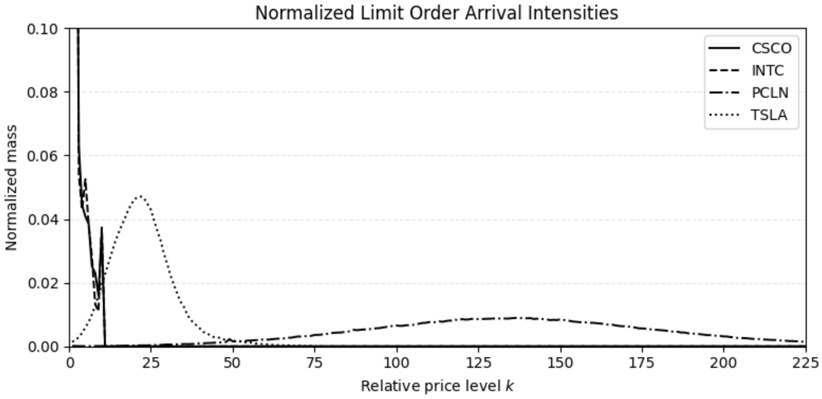

Figure 3.1: Normalized LO arrival intensities by relative price for all stocks.

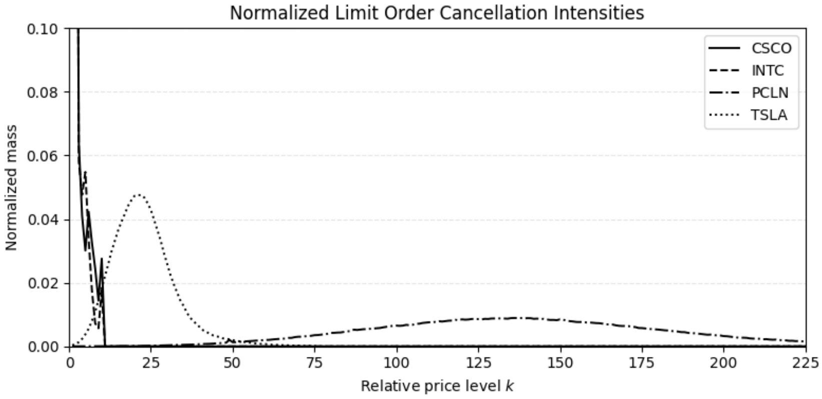

Figure 3.2: Normalized LO cancellation intensities by relative price for all stocks.

Highlights of the estimated parameters. (**Table 3.2** and **Figures 3.1, 3.2**)

- **Activity.** Average total event intensity Λ is highest for INTC and CSCO, lowest for PCLN, with TSLA in between. This indicates markedly higher microstructural activity in INTC and CSCO.
- **Effective depth.** The effective book depth differs across stocks: INTC and CSCO require shallow books ($L = 6$ and $L = 5$, respectively), whereas PCLN and TSLA require deeper books ($L = 78$ and $L = 18$, respectively) to capture trading activity.
- **Passive vs. aggressive flow.** For all stocks, MO intensity γ is approximately one order of magnitude below LO arrival and cancellation intensities λ, $\rho^{\mathrm{a}}+\rho^{\mathrm{b}}$. By event counts, MOs comprise only between 2% and 3% of events. Activity is therefore dominated by passive additions and cancellations.
- **Arrival–cancellation alignment.** Arrival and cancellation shapes closely mirror each other across stocks, reflecting a high-churn regime in which posted liquidity is frequently refreshed.
- **Front vs. deep book.** Intensity-mass varies between stocks: CSCO and INTC concentrate mass within the first few ticks; PCLN shows relatively mass at the best quotes with broader dispersion across depths; TSLA intensities rise with depth, peak, and then taper.

Using the calibrated parameters, we simulate 250,000 market trajectories for each stock, each represented by $N+1 = 129$ snapshots over $[0, T]$. At each t_n, we evolve the state to t_{n+1} using **Algorithm 1**. Bid and ask queues are initialized to the average estimated queue sizes; the mid-price is set to the closing mid-price from the 2015-03-31 15:30 snapshot; the bid-ask spread is set to the spread of the average queues. From each trajectory, we store the features described in Subsection 2.3.4.

3.2 VWAP Policy Evaluation

We present the training results for each stock. For comparability, hedge (execution quantities) and order allocations are divided by the target quantity. The training loss and wealth are divided by the target quantity and initial mid-price.

The target quantity Q_0 is set to 25% of the expected MO volume over $[0, T]$. MO arrivals (buy and sell) have intensity 2γ; MO log-volumes have location and scale parameters μ_{M} and σ_{M}, respectively. Therefore, the expected MO daily volume can be computed as

$$\mathbb{E}\big[\text{MO volume over } [0, T]\big] \; = \; 2T\,\gamma\, e^{\mu_{\mathrm{M}}+\frac{1}{2}\sigma_{\mathrm{M}}^2}. \tag{3.1}$$

Table 3.3 shows each stock's average daily MO volume and target quantity.

Parameter	CSCO	INTC	PCLN	TSLA
Avrg. traded vlm.	6,462,018	5,163,794	127,056	662,124
Target quantity	1,615,505	1,290,949	31,764	165,531

Table 3.3: Average daily traded volume and target quantity per stock.

Training loss. The training loss per epoch exhibits two phases:

- **Phase 1 (pre-elbow):** rapid reduction of obvious errors (e.g., avoiding costly MOs and limiting inventory exposure).
- **Phase 2 (post-elbow):** slower, incremental improvements.

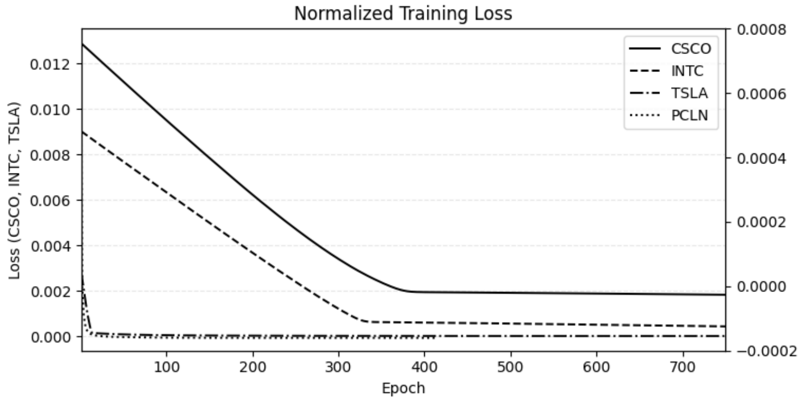

Figure 3.3: Normalized training loss for all stocks.

Wealth Distribution. Applying the learned policies to the training sets yields wealth distributions with slightly positive sample means. This is consistent with agents' tendency to accumulate inventory near the market close (see the discussion below on inventory evolution).

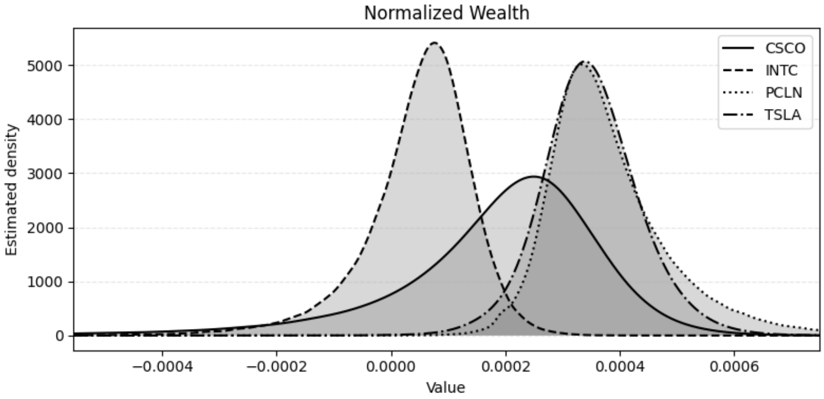

Figure 3.4: Normalized wealth for all stocks.

Parameter	CSCO	INTC	PCLN	TSLA
Expctd. val.	2.9135	0.9651	3.9293	3.7630
Stndrd. dev.	3.7090	1.1593	1.3004	3.7045
RU measure	16.3746	3.3486	-1.5212	-0.1556

Table 3.4: Wealth distribution results for all stock (scaled by 10^4).

Hedge evolution. All stocks except PCLN build inventory uniformly throughout the session and finish with a mean terminal inventory close to the target. This result is coherent with the Santa Fe model's treatment of MOs. PCLN differs for two reasons: (i) late-session LOs at deep relative prices can improve the terminal mark-to-market of the long leg, causing the policy to add size near the close and push mean inventory above target; (ii) PCLN's wider LO intensities profile yields more dispersed fill levels and, consequently, more dispersed final inventories.

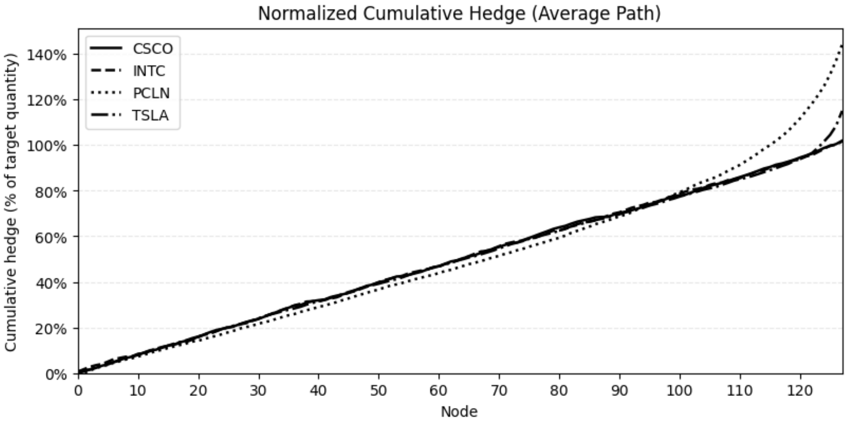

Figure 3.5: Average path of the normalized cumulative hedge for all stocks.

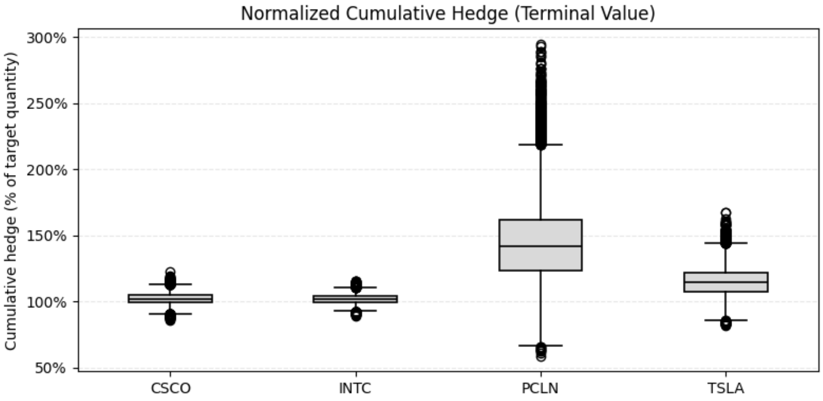

Figure 3.6: Terminal value of the normalized cumulative hedge for all stocks.

Average order allocation. The Santa Fe model is stationary; combined with the risk-averse RU objective penalized with inventory deviations, this favors near-uniform participation and discourages path-dependent swings. Consequently, the learned policy reacts only weakly to the market features, producing relatively stable order allocation throughout the session.

We next discuss asset-specific results for the (path-averaged) learned policies.

CSCO

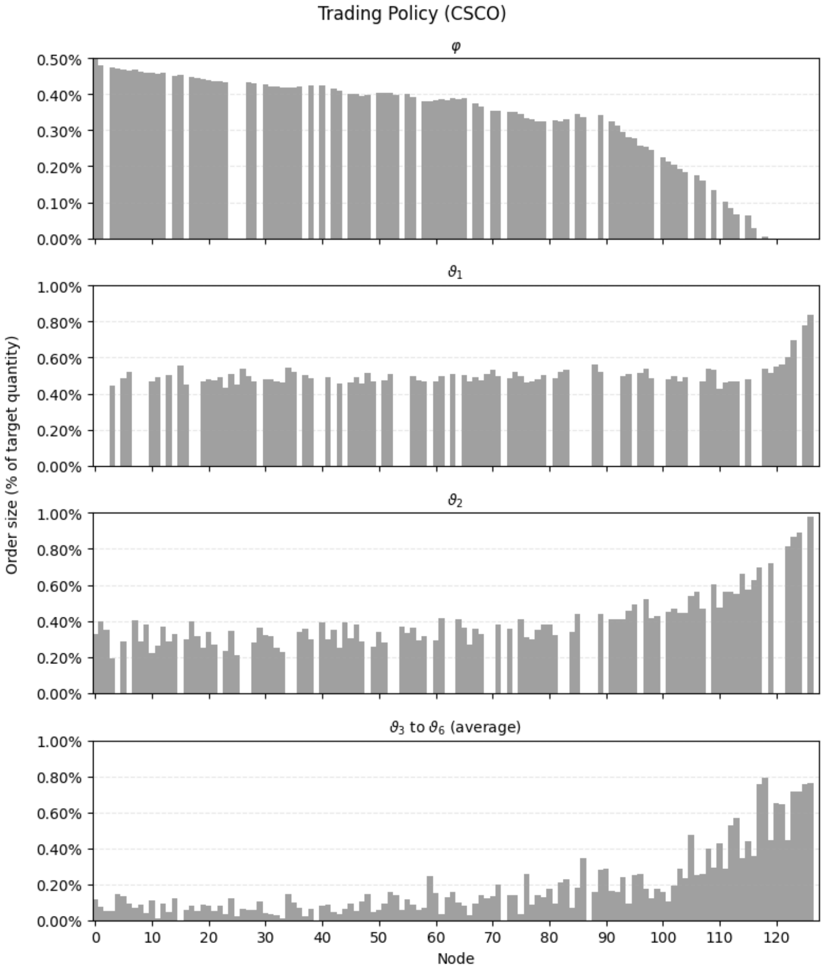

Figure 3.7: Trading policy for CSCO.

The learned policy for CSCO begins with a balanced mix of MOs and LOs and shifts near the end toward an exclusively passive strategy:

- **Early phase ($n \approx$ 0–90).** MO usage starts around 0.5% of Q_0 and decreases toward 0.3%. Level-1 and level-2 LO allocations remain stable between 0.4% and 0.5% per decision step. Posting deeper in the book appears sporadically (around 0.1%) and gradually rises toward 0.3%.
- **Middle phase ($n \approx$ 90–115).** MO taper off to essentially zero. Level-1 LOs remain stable at approximately 0.5%, while level-2 allocations increase to about 0.7%. Deeper-book placements intensify, reaching roughly 0.5%.
- **Late phase ($n \approx$ 115–127).** MO disappear entirely. All passive placements increase markedly, with allocations across levels rising to the 0.8%–1.0% range.

INTC

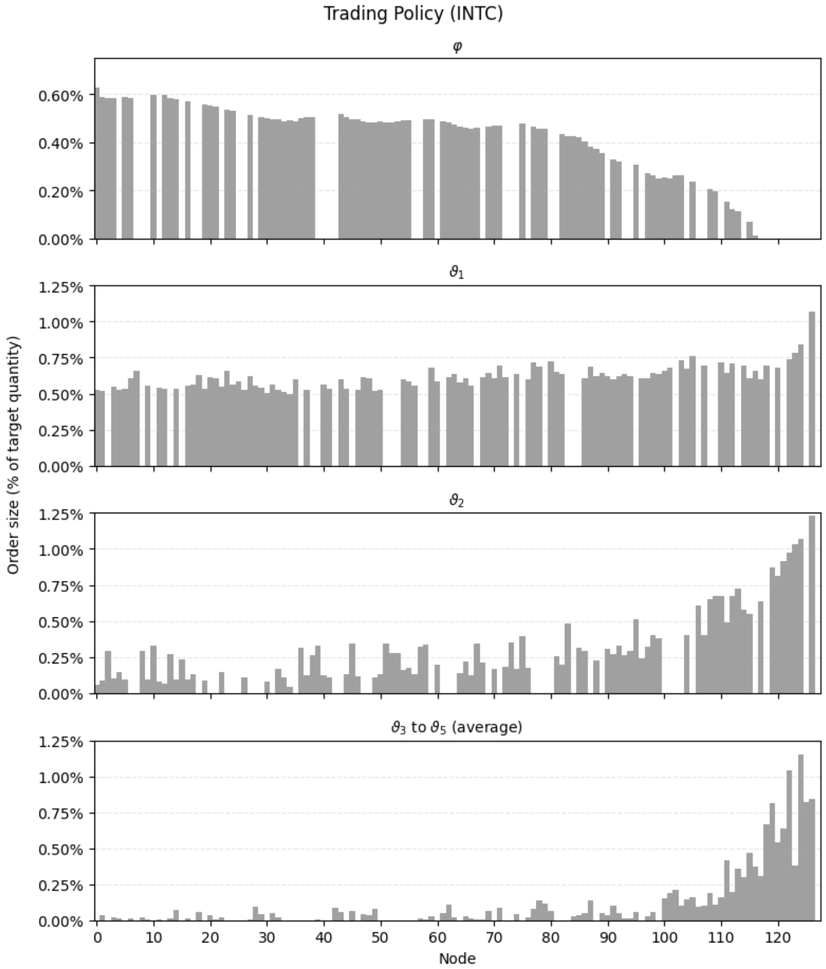

Figure 3.8: Trading policy for INTC.

The learned policy for INTC is characterized by early use of MOs and shallow LOs, followed by a gradual shift toward deeper-book posting:

- **Early phase ($n \approx 0$–80).** MO allocation is stable between 0.4% and 0.6% of Q_0 per decision step, accompanied by level 1 LOs increasing from 0.5% to 0.75%. Level 2 submissions occur intermittently, occasionally reaching 0.35%, while deeper-book allocations remain negligible.
- **Middle phase ($n \approx 50$–115).** MO usage declines toward zero. Level 1 LOs remain active at roughly 0.75%, while deeper-book allocations begin to increase, reaching approximately 0.75%.
- **Late phase ($n \approx 115$–127).** MOs vanish entirely. Level 1 LO allocation increases modestly to about 1.0%. Deeper-level posting expands markedly, rising to between 1.0% and 1.1% toward the end of the trading day.

PCLN

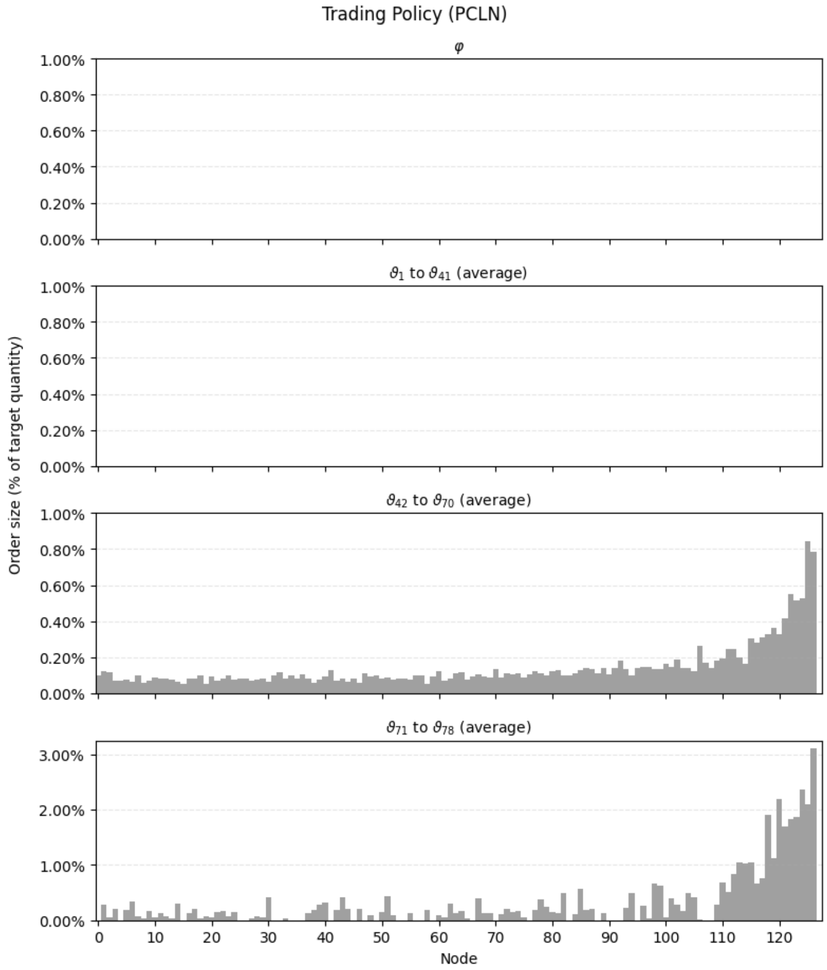

Figure 3.9: Trading policy for PCLN.

The learned policy for PCLN is purely passive and allocated predominantly at deeper book levels, with order sizes increasing over time:

- **Early–mid phase ($n \approx 0$–110).** Neither MOs nor shallow LOs are used. Mid-book posting begins around 0.1% of Q_0 per decision step and rises to approximately 0.2%, while deep-level placements appear only sporadically.
- **Late phase ($n \approx 110$–127).** Both MOs and shallow LOs remain unused. Attention shifts entirely toward deeper levels: mid-book allocations increase from roughly 0.25% to 0.8% per step, while the deepest levels dominate, reaching up to about 3.25% near the end.

TSLA

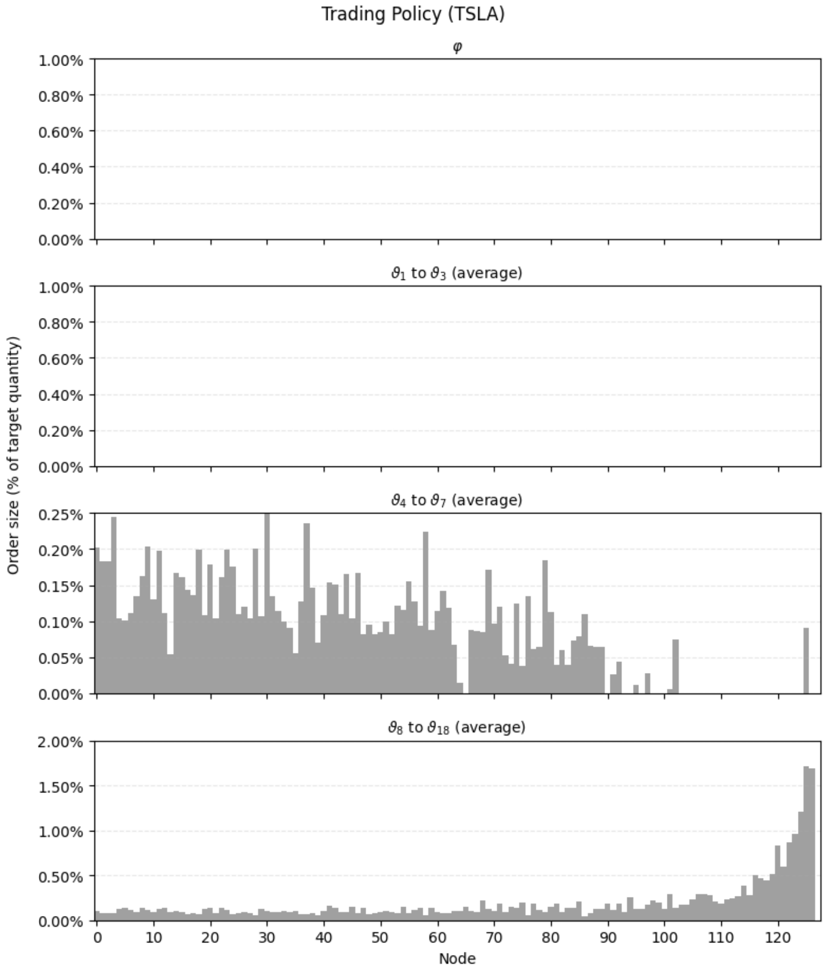

Figure 3.10: Trading policy for TSLA.

The learned TSLA policy is purely passive, with a mid-book bias:

- **Early phase ($n \approx$ 0–80).** No MOs or level-1 LOs are submitted. Mid-book LO placement fluctuates between 0.05% and 0.25% of Q_0 per decision step, while deep-book posting remains stable at roughly 0.1%.
- **Middle phase ($n \approx$ 80–100).** MO usage and level-1 LO posting remain absent. Mid-book allocation gradually declines toward zero, and deep-book LO size increases to approximately 0.2% per step.
- **Late phase ($n \approx$ 100–127).** The strategy remains fully passive. Deep-book posting intensifies markedly, rising to nearly 1.75% of Q_0 per step near the end.

Chapter 4

Conclusion

This thesis examined whether DH can learn economically meaningful VWAP risk-hedging policies in LOB markets. We built an event–driven, parametric market simulator and calibrated model parameters for several stocks. Using simulated market trajectories, we trained a neural network policy to allocate between MOs and LOs across multiple relative prices over one full trading day. The central finding is that the learned DH policies are regime–aware and interpretable. Their structure is related to the shape of LO arrivals and cancellation intensities in the Santa Fe model. Below, we summarize the key findings and outline future research directions.

Reproducibility. All code and processed data used in this project, excluding raw LOBSTER data due to licensing, are available in the following GIT Hub repository: https://github.com/mikegeiser/deep_hedging_VWAP

4.1 Findings

The estimated Santa Fe parameters explain cross-asset differences in the learned policies. The location of mass in $\{\lambda^{(k)}\}$ and $\{\rho^{(k)}\}$ predicts the order-type mix: shallow concentration supports shallow LO posting and MO usage, while concentration at deeper levels favors purely passive, deep LO placement:

- **Tight-spread books suggest early aggressiveness.** CSCO and INTC exhibit high activity near the top of the book: roughly 80% of the mass of $\{\lambda^{(k)}\}$ and $\{\rho^{(k)}\}$ lies within the first three levels. This implies a smaller cost differential between MOs and shallow LOs. Consistently, the learned policies begin with meaningful MO usage accompanied by shallow posting.
- **Spread-out books suggest purely passive deep placement.** PCLN exhibits activity concentrated deep in the book: about 80 % of the mass of $\{\lambda^{(k)}\}$ and $\{\rho^{(k)}\}$ lies within $80 \leq k \leq 200$. This corresponds to high MO costs and renders shallow LOs suboptimal, favoring passive allocation at deep levels. Consistently, the learned strategy avoids MOs and shallow/mid-book posting, progressively increasing deep LO sizes.
- **Intermediate-structure books suggest passive allocation at mid levels.** TSLA sits between the extremes: about 80 % of the mass is

concentrated within $12 \leq k \leq 35$. The policy allocates LOs only, with an initial mid-book bias and a gradual drift toward deeper placement.

In all cases, a late-session ramp-up in LO posting emerges, consistent with a higher probability of execution below the closing mid-price.

4.2 Future Work

Market model realism. In the traditional Santa Fe model, event intensities are fixed through the trading session. To better reproduce intraday stylized facts, one can introduce state–dependence in order intensities and fit time/price–dependent distributions for order volumes. A more advanced approach is to adopt non-parametric, data–driven LOB generators trained on real streams (see [6] for an autoregressive recurrent neural network factoring event type, price level, size, and interarrival time).

Compressed input features. The current feature set includes the raw per–side queue vectors. A compact summary could reduce dimensionality while preserving shape information. Options include (i) parametric fits to arrival and cancellation profiles or (ii) curve signatures that encode the book's geometry (see [13], where the authors extract signature path features from LOB data and compress them for training). Additional state variables (e.g., posted and canceled liquidity, signed trade flow, realized volatility) could further improve the information stream.

Endogenous market impact. A more realistic setting, particularly relevant for desks trading directly on LOBs, is to allow the agent's orders to affect the book and, through impact, the price dynamics. Within the DH framework, this can be implemented by simulating exogenous shocks and propagating the state with control-dependent transitions. For instance, suppose that we model prices via a GBM. On a grid $t_n = n\Delta t$ with i.i.d. $Z_{n+1} \sim \mathcal{N}(0,1)$, a control-dependent GBM takes the form

$$S_{n+1} = S_n \exp\Big\{\big(\mu(\delta_n) - \tfrac{1}{2}\sigma^2(\delta_n)\big)\Delta t \;+\; \sigma(\delta_n)\sqrt{\Delta t}\, Z_{n+1}\Big\}, \tag{4.1}$$

where $\mu(\delta_n)$ and $\sigma(\delta_n)$ encode drift/volatility changes due to the agent's trade δ_n at step n. For example, large trades may increase σ or induce a short-lived drift via μ. The simulated shocks $\{Z_n\}$ are used as training data; the model maps shocks and controls to terminal wealth. Similar techniques might be applied to a LOB parametric model.

Appendix A

Estimated Santa Fe Model Parameters

Let M denote the number of trading days in the sample, and let $\mathrm{MO}^{\pm}$, $\mathrm{LO}_k^{\pm}$, and $\mathrm{CXL}_k^{\pm}$ denote, respectively, MOs submissions, LO arrivals at relative price $k \in \{1, \ldots, K\}$, and LO cancellations at relative price k, on the buy $(+)$ and sell $(-)$ sides. Denote by $N_T^m(\cdot)$ the total count of a given event type on day $m \in \{1, \ldots, M\}$ over the window $[0, T]$. The intensity estimators are

$$\gamma = \frac{1}{M} \frac{\sum_{m=1}^{M} \big(N_T^m(\mathrm{MO}^-) + N_T^m(\mathrm{MO}^+)\big)}{2T}, \tag{A.1a}$$

$$\lambda^{(k)} = \frac{1}{M} \frac{\sum_{m=1}^{M} \big(N_T^m(\mathrm{LO}_k^-) + N_T^m(\mathrm{LO}_k^+)\big)}{2T}, \tag{A.1b}$$

$$\rho^{(k)} = \frac{1}{M} \frac{\sum_{m=1}^{M} \big(N_T^m(\mathrm{CXL}_k^-) + N_T^m(\mathrm{CXL}_k^+)\big)}{2V^{(k)}T}, \tag{A.1c}$$

where

$$V^{(k)} = \frac{1}{M} \frac{\sum_{m=1}^{M} \int_0^T \big(a_{t,m}^{(k)} + b_{t,m}^{(k)}\big)\, dt}{2T} \tag{A.2}$$

is the time–weighted average queue size at relative level k (per side). To estimate $V^{(k)}$, we use the stream of depth snapshots over the trading window. For CSCO and INTC, the LOBSTER order–book files report the top ten price levels at each event (mapped to relative indices), which suffices for these names; $V^{(k)}$ is then obtained directly from snapshots. For PCLN and TSLA, we reconstruct snapshots by initializing the book with the first observed ten levels and updating it event–by–event using the corresponding message file, following **Algorithm 1**; the resulting snapshot sequence is used to compute $V^{(k)}$.

To estimate the parameters for order volumes, we assume log–normal sizes and aggregate daily log–moments. Let $\{V_i^m\}_{i=1}^{n_m}$ be the set of observed order volumes on day m. The pooled estimators are

$$\mu_{\log V} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{n_m} \log V_i^m}{\sum_{m=1}^{M} n_m}, \tag{A.3a}$$

$$\sigma_{\log V}^2 = \frac{\sum_{m=1}^{M} \sum_{i=1}^{n_m} (\log V_i^m)^2}{\sum_{m=1}^{M} n_m} - \mu_{\log V}^2, \tag{A.3b}$$

where n_m is the number of valid volume observations on day m.

Since relative prices may extend far into the book, a truncation rule is applied to control the number of levels. The smallest K is selected such that the cumulative intensity, aggregated over relative prices from 1 up to 300, accounts for at least 99% of the total observed intensity. Residual flow beyond K is pooled at the book boundary in the simulator. In the tables below we report parameters up to the first 40 relative prices; complete tables are provided in the GitHub repository.

CSCO

k	$V^{(k)}$	$\lambda^{(k)}$	$10^4 \cdot \rho^{(k)}$
1	8,926	2.29687448	2.42071557
2	14,750	1.16566609	0.87280240
3	17,404	0.30402923	0.17409926
4	17,238	0.21531379	0.11333774
5	16,353	0.19808246	0.08771246
6	13,337	0.18688318	0.15107668
7	10,467	0.12195728	0.14872771
8	9,535	0.11200778	0.12221726
9	8,948	0.07733855	0.07639037
10	7,764	0.17574267	0.16921272

Table A.1: CSCO depth–wise parameters: average queue size $V^{(k)}$, limit order arrival intensities $\lambda^{(k)}$, and cancellation intensities $\rho^{(k)}$ (scaled by 10^4).

INTC

k	$V^{(k)}$	$\lambda^{(k)}$	$10^4 \cdot \rho^{(k)}$
1	4,794	3.08815739	5.96026109
2	8,207	1.69783366	2.19539475
3	10,009	0.35684426	0.36075563
4	10,288	0.28474499	0.22858504
5	9,682	0.34807294	0.28612336
6	8,173	0.25343393	0.30315003
7	5,800	0.18587928	0.42628265
8	5,039	0.08802823	0.17943223
9	4,812	0.07266807	0.11956041
10	4,377	0.24736587	0.35340851

Table A.2: INTC depth–wise parameters: average queue size $V^{(k)}$, limit order arrival intensities $\lambda^{(k)}$, and cancellation intensities $\rho^{(k)}$ (scaled by 10^4).

PCLN

k	$V^{(k)}$	$\lambda^{(k)}$	$10^4 \cdot \rho^{(k)}$
1	1	0.00014986	0.20012195
2	2	0.00009728	0.09818079
3	1	0.00007948	0.11931517
4	1	0.00007327	0.15055394
5	1	0.00009439	0.07798859
6	2	0.00009563	0.07671823
7	3	0.00007948	0.04301145
8	3	0.00010018	0.06415706
9	3	0.00009687	0.08253847
10	4	0.00011591	0.08669407
11	4	0.00011219	0.07390579
12	5	0.00012088	0.08329535
13	6	0.00011177	0.05838113
14	6	0.00012171	0.07075785
15	6	0.00014862	0.07307683
16	7	0.00016435	0.09343103
17	8	0.00017635	0.08920254
18	10	0.00019788	0.08881819
19	10	0.00024011	0.11010257
20	11	0.00028854	0.10637789
21	13	0.00027447	0.09474769
22	14	0.00032125	0.10580016
23	14	0.00032166	0.12911865
24	15	0.00032870	0.12353589
25	17	0.00041936	0.12967165
26	20	0.00041108	0.10922168
27	20	0.00043881	0.12715477
28	23	0.00048601	0.12771892
29	26	0.00054231	0.12425890
30	27	0.00066609	0.14136118
31	30	0.00060275	0.11935806
32	34	0.00065946	0.12155469
33	37	0.00069714	0.11182409
34	39	0.00074433	0.11645328
35	43	0.00078738	0.12235030
36	49	0.00078448	0.10965036
37	50	0.00078614	0.11018034
38	55	0.00085693	0.10897912
39	60	0.00097367	0.11442788
40	64	0.00107013	0.11535486
⋮	⋮	⋮	⋮
295	41	0.00022065	0.05999261

Table A.3: PCLN depth–wise parameters: average queue size $V^{(k)}$, limit order arrival intensities $\lambda^{(k)}$, and cancellation intensities $\rho^{(k)}$ (scaled by 10^4).

TSLA

k	$V^{(k)}$	$\lambda^{(k)}$	$10^4 \cdot \rho^{(k)}$
1	4	0.00326710	1.68232230
2	18	0.00473630	0.89832582
3	35	0.00664307	0.83860797
4	72	0.00979632	0.72252601
5	124	0.01381437	0.67150471
6	199	0.01770906	0.61263841
7	307	0.02327206	0.57014914
8	408	0.02970566	0.58826833
9	523	0.03735966	0.61082853
10	659	0.04526329	0.61046056
11	781	0.05249089	0.62095754
12	890	0.06052202	0.64046416
13	975	0.06812179	0.67424273
14	1066	0.07555638	0.68528195
15	1149	0.08206201	0.69790085
16	1204	0.08765400	0.71544337
17	1249	0.09289742	0.73154860
18	1318	0.09910291	0.73585987
19	1350	0.10347036	0.74733425
20	1396	0.10740934	0.74542803
21	1473	0.10876842	0.71061891
22	1501	0.10933598	0.69709813
23	1538	0.10775128	0.66946608
24	1570	0.10381893	0.63104485
25	1614	0.09977397	0.58529327
26	1623	0.09177637	0.53968460
27	1641	0.08416667	0.48906053
28	1631	0.07710051	0.44862499
29	1594	0.06751987	0.40400004
30	1507	0.06037216	0.37802930
31	1474	0.05171386	0.33423511
32	1438	0.04517056	0.29929053
33	1367	0.03892739	0.27432036
34	1295	0.03277033	0.24527683
35	1231	0.02936413	0.22721264
36	1152	0.02372288	0.20235128
37	1095	0.01968662	0.18101710
38	1008	0.01734848	0.17324636
39	942	0.01521982	0.16195962
40	887	0.01297897	0.14817614
$\vdots$	$\vdots$	$\vdots$	$\vdots$
75	159	0.00026205	0.01739388

Table A.4: TSLA depth–wise parameters: average queue size $V^{(k)}$, limit order arrival intensities $\lambda^{(k)}$, and cancellation intensities $\rho^{(k)}$ (scaled by 10^4).

Bibliography

[1] Frédéric Abergel, Marouane Anane, Anirban Chakraborti, Aymen Jedidi, and Ioane Muni Toke. *Limit Order Books*. 2016. doi: 10.1017/CBO9781316683005.

[2] Jean-Philippe Bouchaud, Julius Bonart, Jonathan Donier, and Martin Gould. *Trades, Quotes and Prices: Financial Markets Under the Microscope*. 2018. doi: 10.1017/9781108547435.

[3] Hans Buehler, Lukas Gonon, Josef Teichmann, and Ben Wood. Deep hedging. 2019. doi: 10.1080/14697688.2019.1571683.

[4] WooJae Byun, Bumkyu Choi, Seongmin Kim, and Joohyun Jo. Practical application of deep reinforcement learning to optimal trade execution. 2023. doi: 10.3390/fintech2030023.

[5] Rémi Genet. Recurrent neural networks for dynamic vwap execution: Adaptive trading strategies with temporal kolmogorov-arnold networks. 2025. doi: 10.48550/arXiv.2502.18177.

[6] Hanna Hultin, Henrik Hult, Alexandre Proutiere, Samuel Samama, and Ala Tarighati. A generative model of a limit order book using recurrent neural networks. 2023. doi: 10.1080/14697688.2023.2205583.

[7] Michaël Karpe, Jin Fang, Zhongyao Ma, and Chen Wang. Multi-agent reinforcement learning in a realistic limit order book market simulation. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021. doi: 10.1145/3383455.3422570.

[8] Yao Li, Wenbo Huang, Xuejun Yang, Qinglong Gu, and Guoqiang Li. Hierarchical deep reinforcement learning for VWAP strategy optimization in limit order book markets. 2022. doi: 10.1016/j.eswa.2022.117607.

[9] Siyu Lin and Peter A. Beling. An end-to-end optimal trade execution framework based on proximal policy optimization. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*, 2020. doi: 10.24963/ijcai.2020/627.

[10] LOBSTER Data Service. Lobster: Limit order book reconstruction system, 2013. URL https://lobsterdata.com.

[11] Alexander J. McNeil, Rüdiger Frey, and Paul Embrechts. *Quantitative Risk Management: Concepts, Techniques, and Tools*. 2010.

[12] R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. 2000. doi: 10.21314/JOR.2000.038.

[13] Thendo Sidogi, Wilson Tsakane Mongwe, Rendani Mbuvha, Peter Olukanmi, and Tshilidzi Marwala. A signature transform of limit order book data for stock price prediction. 2023. doi: 10.1109/ACCESS.2023.3293064.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.