

RESEARCH ARTICLE

Reassessing Reinforcement Learning for Optimal Trade Execution

Matteo Tosi¹, Enrico Barbierato^{1,*}  and Alice Gatti¹ 

¹Department of Mathematics and Physics, Catholic University of the Sacred Heart, Italy

Abstract: Recent research has explored reinforcement learning (RL) for optimal trade execution because learning-based policies can adapt execution decisions throughout the liquidation horizon rather than following a predetermined schedule. However, implementation shortfall depends not only on execution decisions but also on realized intraday price paths, stochastic training effects, and the composition of the out-of-sample evaluation set. This study reassesses the performance of RL-based execution policies against Time-Weighted Average Price (TWAP) using a microstructure-aware simulation framework and a statistically controlled evaluation protocol. All strategies are evaluated on identical fixed out-of-sample intraday price paths, enabling paired comparisons that separate execution effects from market drift. Transaction costs are modeled explicitly, and robustness is assessed across multiple training seeds. Unconstrained RL policies frequently generate unstable liquidation schedules and fail to provide reliable improvements over TWAP. To address this instability, this study introduces a residual formulation in which the learning agent makes bounded adjustments around the TWAP schedule. The residual constraint eliminates extreme execution patterns and improves policy stability and interpretability. Across up to 1,000 test episodes, residual RL remains close to TWAP performance without achieving statistically reliable improvements. A drift–friction decomposition further shows that realized price movements dominate outcome variability, while frictional costs remain broadly similar across strategies.

Keywords: optimal trade execution, reinforcement learning, algorithmic trading

1. Introduction

Large orders are seldom executed as a single trade. A one-shot execution can consume visible depth, widen the effective spread, and reveal information about the trader's intentions. In practice, parent orders are split into smaller child orders and executed over a pre-specified horizon. The resulting execution schedule determines how the trader balances inventory risk against transaction costs. Executing early reduces exposure to future price movements but typically increases spread and market impact costs, while slower execution reduces immediate pressure on the order book at the expense of prolonged exposure. Implementation shortfall (IS) provides a natural measure of this trade-off by comparing the value of the order at the arrival price with the realized execution proceeds [1].

Optimal execution has traditionally been studied within model-based frameworks that combine market microstructure, transaction costs, and inventory risk. Comprehensive treatments such as Cartea et al. [2] formulate execution as a stochastic control problem, where trading decisions are taken under explicit assumptions about market impact and liquidity. More recent work extends this perspective by introducing time-varying liquidity and stochastic impact, leading to execution policies that adapt to evolving market conditions rather than relying on static cost

specifications [3]. These models not only highlight the importance of explicitly modeling market frictions but also rely on simplifying assumptions that may not fully capture the complexity of intraday trading.

In real markets, liquidity is highly dynamic, prices are discrete, and volatility and order-book conditions fluctuate over short horizons. At the same time, empirical evidence suggests that short-term price predictability is weak, so that improvements in performance are more likely to come from execution quality than from directional forecasting [4]. This motivates the search for more flexible approaches that can adapt to complex market conditions without relying on explicit parametric models.

Reinforcement learning (RL) offers one such approach. By framing execution as a sequential decision problem, RL methods allow policies to adapt dynamically to the observed market state. Recent studies report that RL-based execution rules can outperform simple benchmarks such as Time-Weighted Average Price (TWAP) or analytically derived schedules in simulated or empirical settings [5–8]. These results have generated interest in data-driven execution policies that can adjust trading behavior in real time.

However, these findings raise an important methodological question. In execution problems, observed performance depends not only on the trading strategy but also on the realized price path during the execution horizon. A policy may appear to outperform a benchmark simply because it is evaluated on favorable episodes. Small test sets, rolling splits, or unpaired comparisons

*Corresponding author: Enrico Barbierato, Department of Mathematics and Physics, Catholic University of the Sacred Heart, Italy. Email: enrico.barbierato@unicatt.it

can therefore attribute market drift to execution skill. Additional sources of variability arise from stochastic training and from the sensitivity of learned policies to initialization and exploration noise. In unconstrained settings, RL agents may also produce economically implausible schedules, such as extreme front-loading or back-loading of trades.

This paper revisits RL execution under a controlled evaluation framework designed to isolate the effect of the trading policy. All strategies are tested on the same fixed out-of-sample intraday price paths, and comparisons are conducted on a paired, episode-by-episode basis. Performance is aggregated across multiple training seeds, and transaction costs are modeled explicitly. The central question is whether, under sparse information and realistic frictions, a learning-based policy can produce a statistically reliable improvement over a simple time-based benchmark.

To further stabilize the learning problem, the policy is restricted to bounded residual adjustments around a TWAP schedule. This prevents extreme execution patterns and maintains economic interpretability, while still allowing the agent to modify the timing of trades. The residual formulation separates two distinct questions: whether RL can eliminate pathological schedules, and whether it can deliver consistent improvements in execution performance.

The empirical analysis addresses three questions:

- 1) Do unconstrained RL execution policies outperform TWAP under paired out-of-sample evaluation?
- 2) Does a residual formulation produce more stable and economically interpretable schedules?
- 3) Once this constraint is imposed, does RL achieve statistically reliable improvements, or does it remain close to benchmark performance?

The remainder of the paper is organized as follows. Section 2 defines the execution problem, performance metrics, and cost decomposition. Section 3 reviews related work. Section 4 presents the data, models, and empirical results. Section 5 discusses the interpretation of the findings, and Section 6 concludes.

2. Background and Preliminaries

Execution of a single asset over a fixed intraday horizon, split into T decision intervals of equal length, is considered. Let $\{P_0, P_1, \dots, P_T\}$ denote the observed midprices at the interval boundaries, where P_0 is the arrival price. The price paths used in the experiments come directly from historical intraday data. A parametric price process is neither fitted nor assumed; in particular, no diffusion, martingale, or stationarity assumption is made. The realized paths can therefore contain drift, volatility clustering, short-lived trends, and microstructure noise, all of which are common in intraday data [9].

For each interval $t \in 0, \dots, T-1$, a representative midprice $\bar{P}_t$ is used. Unless stated otherwise, this is defined as the midpoint of the two boundary prices,

$$\bar{P}_t = \frac{P_t + P_{t+1}}{2}. \quad (1)$$

One could instead use, for example, an arithmetic average over higher-frequency observations inside the interval. For the comparisons below, $\bar{P}_t$ must be constructed deterministically from the realized path, so that every strategy is evaluated against the same price sequence.

Intraday drift can be large relative to the differences produced by alternative execution schedules. If two strategies are

tested on different paths, measured performance differences may reflect favorable or unfavorable market moves rather than execution quality, a known problem in empirical strategy evaluation [10]. For this reason, all comparisons in the paper are made on identical price paths.

An agent must liquidate an initial inventory $Q_0 > 0$ over the T intervals. Let $q_t \geq 0$ be the quantity executed during interval t . Inventory evolves as

$$Q_{t+1} = Q_t - q_t, \quad Q_0 \text{ given}, \quad Q_T = 0, \quad (2)$$

where Q_t is the remaining inventory at the beginning of interval t . Feasible schedules satisfy

$$q_t \in [0, Q_t], \quad \sum_{t=0}^{T-1} q_t = Q_0. \quad (3)$$

The terminal constraint $Q_T = 0$ enforces full liquidation, matching common institutional requirements such as end-of-day inventory limits [11].

The TWAP strategy uses the deterministic schedule

$$q_t^{\text{TWAP}} = \frac{Q_0}{T}, \quad t = 0, \dots, T-1, \quad (4)$$

up to integer rounding. TWAP is used as the primary benchmark because it is widely used in practice and provides a simple reference point without requiring a price model.

Execution prices include explicit transaction frictions. For a sell order executed in interval t , the realized execution price is modeled as

$$\hat{P}_t = \bar{P}_t - s - k \phi(q_t; Q_0) + \varepsilon_t, \quad (5)$$

where $s > 0$ is the half-spread, $k > 0$ is a temporary linear impact coefficient, $\phi(\cdot)$ is a normalized size term, and ε_t captures residual execution noise. In the experiments, the following specification is used:

$$\phi(q_t; Q_0) = \frac{q_t}{Q_0}, \quad (6)$$

so that impact increases with the fraction of the full order traded in a single interval. The residual term is modeled as bounded noise,

$$\varepsilon_t \sim \text{Unif}(-\epsilon, \epsilon), \quad (7)$$

with amplitude $\epsilon \geq 0$, denoted `residual_eps` in the implementation. This term represents within-interval execution variation, including queue effects and other unobserved liquidity conditions. When two strategies are compared on the same episode, they use the same noise realization, which keeps the comparison paired.

Given an execution schedule $\{q_t\}_{t=0}^{T-1}$, total cash proceeds are

$$X = \sum_{t=0}^{T-1} q_t \hat{P}_t. \quad (8)$$

The execution-price model in (5) keeps the two frictions needed for the experiments, bid-ask spread costs and temporary impact.

Performance is measured by IS relative to the arrival price P_0 :

$$\text{IS} = Q_0 P_0 - X. \quad (9)$$

For comparison across episodes, IS is also reported in basis points,

$$\text{IS}_{\text{bps}} = \frac{\text{IS}}{Q_0 P_0} \times 10^4. \quad (10)$$

Two details matter for the empirical analysis. First, IS may be negative when the average execution price is above the arrival price, as can happen on days with strong upward drift. Second, because drift can dominate the execution effect itself, paired comparisons on the same paths are used primarily, rather than unpaired averages across different trading days. To separate market drift from execution frictions, IS is decomposed. Define the frictionless proceeds

$$X^{\text{mid}} = \sum_{t=0}^{T-1} q_t \bar{P}_t, \quad (11)$$

and the associated drift component

$$\text{IS}^{\text{drift}} = Q_0 P_0 - X^{\text{mid}} = \sum_{t=0}^{T-1} q_t (P_0 - \bar{P}_t). \quad (12)$$

The remaining component is the frictional cost,

$$\text{IS}^{\text{fric}} = X^{\text{mid}} - X = \sum_{t=0}^{T-1} q_t (\bar{P}_t - \hat{P}_t). \quad (13)$$

By construction,

$$\text{IS} = \text{IS}^{\text{drift}} + \text{IS}^{\text{fric}}. \quad (14)$$

This decomposition is useful in the intraday setting because realized price moves can be larger than the differences between execution policies.

Learning-based policies operate as residual adjustments around a TWAP schedule. This follows recent work on RL-based execution under non-stationary liquidity conditions, while keeping the policy economically interpretable and constrained [12]. Let

$$\tilde{q}_t = q_t^{\text{TWAP}} (1 + \delta_t),$$

where the residual action δ_t is bounded by

$$\delta_t \in [-\varepsilon_\delta, \varepsilon_\delta]. \quad (15)$$

The actual executed quantity is then projected back into the feasible region:

$$q_t = \min\{Q_t, \max\{0, \tilde{q}_t\}\}, \quad (16)$$

with the final interval used to complete liquidation,

$$q_{T-1} = Q_{T-1}. \quad (17)$$

Under this residual formulation, execution is modeled as a finite-horizon Markov decision process. At each step t , the agent observes a state S_t and chooses an action $A_t = \delta_t$. A representative state is

$$S_t = (t/T, Q_t/Q_0, \text{price features at } t), \quad (18)$$

where time and inventory are normalized to improve scale stability across episodes [13].

The reward is chosen to align with IS minimization. The reward is defined as the negative incremental shortfall contribution,

$$R_{t+1} = -\frac{q_t (P_0 - \hat{P}_t)}{Q_0 P_0} \times 10^4, \quad (19)$$

so the undiscounted return satisfies

$$\sum_{t=0}^{T-1} R_{t+1} = -\text{IS}_{\text{bps}}. \quad (20)$$

Maximizing expected return is therefore the same objective as minimizing expected IS. Since the horizon is short and fixed, the undiscounted objective is used:

$$\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T-1} R_{t+1} \right], \quad (21)$$

where π denotes the execution policy. Policies are trained with standard policy-gradient methods, specifically Proximal Policy Optimization (PPO) [14, 15].

3. Related Work

In the scientific literature, optimal execution is often treated as a finite-horizon control problem. The trader has to liquidate a parent order while balancing price risk, liquidity constraints, and market impact. Sadoghi and Vecer [16], for example, study liquidation in illiquid markets. Their setting is relevant here because liquidity constraints can change the optimal schedule itself, rather than simply adding a penalty to a strategy that would otherwise be unchanged. Timing and cost specification are therefore linked from the start.

Not all related work studies the full liquidation problem directly. Some papers focus on components that enter the execution decision. Maglaras et al. [17] use deep learning on order-book data to estimate limit-order fill probabilities. Such a model does not tell a trader how to split and time the whole parent order, but it can affect whether the trader posts passively, crosses the spread, or changes child-order size. This is one plausible role for learning in execution: estimating local market conditions while leaving the broader execution rule more structured.

Simulation is another way to move beyond closed-form execution models. Roch [18] studies liquidation in a simulated limit order book using neural-network methods. The simulator can include microstructure effects that are awkward to represent analytically. Once the policy is learned inside a simulated market, its usefulness depends on whether that market captures the mechanisms that matter outside the simulator. A policy can look strong in-sample and still be fragile if the simulated book omits important behavior.

A separate part of the literature treats liquidity and impact as endogenous rather than fixed background objects. Fu et al. [19] study liquidation with endogenous entry and exit in a mean-field model, while Cordoni and Lillo [20] derive transient impact from Nash equilibria in execution games. These papers are not direct benchmarks for the single-agent experiments below. They are relevant because they show what is missing from simpler execution environments: market impact may be produced by strategic interaction among traders, not only by an imposed cost function.

RL has also been applied to nearby trading problems. Schnaubelt [21] applies deep RL to cryptocurrency limit-order placement, where the agent must handle fill uncertainty,

transaction costs, and noisy microstructure signals. Parent-order liquidation is a different problem, but the practical difficulties are similar. Policy behavior depends heavily on the environment, training may be unstable, and the resulting strategy is often less transparent than a rule-based benchmark.

Deep hedging gives a useful comparison from another financial control problem. Buehler et al. [22] use neural networks to learn hedging policies under market frictions. The objective is not execution, but the methodological point carries over. The performance of a learned financial policy is tied to the chosen state variables, the cost specification, and the sampling procedure. These are modeling choices, not minor implementation details.

Mascioli et al. [23] introduce a simulation environment for trading agents under stylized microstructure dynamics. Such platforms are useful because matching rules, randomness, and agent interaction are made explicit. They also make evaluation less innocuous than it may appear. Changing the data-generating process, the seed protocol, or the matching mechanism can change the ranking of learned agents.

Evidence from other market domains helps frame what one should expect from RL in execution. Cheng et al. [24] review deep RL and evolutionary game-theoretic methods in user-side electricity markets, where agents operate in high-dimensional, interactive environments. Equity execution, at least in the single-agent liquidation setting considered here, is narrower. Adaptive methods are more compelling when the state contains exploitable structure or when strategic interaction is central. A sparse-state liquidation problem is therefore a relatively demanding test of what RL adds beyond a simple schedule.

Several recent execution papers report that RL policies can outperform TWAP or analytical schedules. Zhang et al. [25], for instance, explicitly investigate the generalization of RL policies for trade execution and the risk of overfitting to limited market contexts. Other studies based on value-function methods or hierarchical policies also find gains under specific experimental designs. Those results motivate the comparison in this paper, but they also make the evaluation protocol important. Without paired test paths, a sufficiently large test set, and checks across random seeds, apparent gains may come from drift or sampling variation rather than better execution decisions.

In this paper, policies are evaluated on the same fixed out-of-sample intraday paths, compared through paired episode-level differences, and trained across multiple seeds with explicit transaction costs. Under this protocol, the RL policies considered here do not deliver statistically reliable improvements over TWAP.

4. Models and Experimental Design

Each policy is evaluated with the same transaction-cost model and on the same realized intraday price paths. In this way, the price path dominates the measured outcome. A schedule tested during favorable drift may look effective even when the rule itself has no execution advantage.

The agent does not observe order-book imbalance, trade flow, volume forecasts, cross-asset variables, or other signals that could reasonably be interpreted as short-term alpha. In the residual specification, the agent also cannot choose an arbitrary liquidation curve. It can only make bounded adjustments around TWAP.

4.1. Experimental rationale

IS changes substantially across days, even for a deterministic schedule. The same rule may be executed through an upward

move, a downward move, or a volatile path with little net direction. If two policies are evaluated on different episodes, the measured gap mixes the schedule effect with the effect of the particular price path. For this reason, comparisons are paired at the episode level. On each out-of-sample path, TWAP and the residual PPO policy face the same prices. When execution noise is included, both policies receive the same noise realization. The comparison is pathwise: only the schedule changes, while the market path is held fixed.

First, the dispersion of TWAP IS on the fixed test set is examined. Residual PPO policies are then trained across random seeds. The resulting schedules and IS values are examined using paired differences and the drift–friction decomposition from Section 2.

4.2. Data description and preprocessing

The data are 1-minute intraday observations for IBM, a liquid US equity listed on the NYSE, from 1998 to 2025. Prices are represented by midquotes, computed from the best bid and best ask at each timestamp. Midquotes are used rather than transaction prices so that bid–ask bounce is not built directly into the comparison between execution schedules.

The sample contains 7,030 trading days, with each day defining one execution episode. Table 1 reports the main summary statistics.

The raw market data are proprietary and cannot be redistributed. The preprocessing choices, simulator structure, and evaluation protocol are reported so that the experiment can be repeated with comparable intraday data.

Table 1
Summary statistics of intraday price paths

Statistic	Value
Number of assets	1 (IBM)
Trading days	7,030
Time span	1998–2025
Sampling frequency	1 minute
Mean intraday return (bps)	0.013
Std. intraday return (bps)	8.99
Mean daily volatility (bps)	7.36

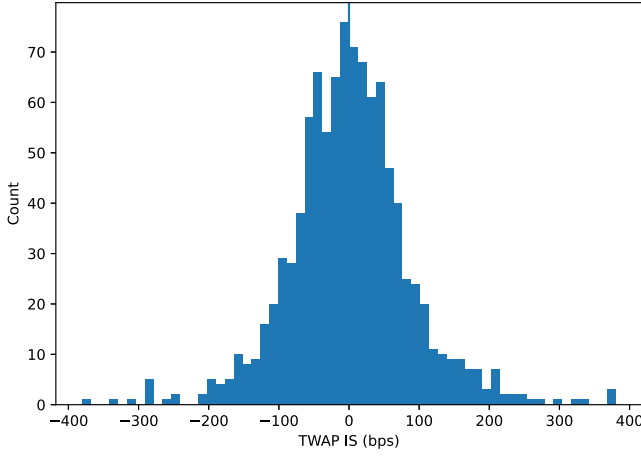
4.3. Baseline model

The baseline policy is TWAP. Given the horizon and initial inventory, the parent order is split evenly across the execution window and fully liquidated by the final step. The rule does not require a volume curve, a price forecast, or an impact calibration.

A Volume-Weighted Average Price (VWAP) schedule would require an assumed or estimated intraday volume profile, and an Almgren–Chriss schedule would require choices for risk aversion, volatility, and impact parameters. As a result, these benchmarks introduce additional assumptions that are not needed for the evaluation question studied here.

Figure 1 reports TWAP IS over 1,000 fixed out-of-sample episodes. The distribution is wide and crosses zero, showing how much the realized intraday path can change the outcome of the same execution rule.

Figure 1
Distribution of TWAP implementation shortfall



4.4. Learning-based execution model

The learning policy is trained using PPO. The action is continuous because the policy adjusts the quantity traded at each decision step. Using PPO avoids imposing a discrete grid of trade sizes.

The residual policy modifies TWAP multiplicatively. At decision step t ,

$$q_t = q_t^{\text{TWAP}} (1 + \delta_t), \quad (22)$$

where δ_t is chosen by the policy and constrained by

$$\delta_t \in [-\varepsilon_\delta, \varepsilon_\delta]. \quad (23)$$

The bound prevents the learned policy from replacing TWAP with a completely different liquidation curve. It also removes the extreme front-loaded and back-loaded schedules that appeared in preliminary unconstrained runs.

The reward is negative IS in basis points. Because the episode return is undiscounted, maximizing reward is equivalent to minimizing IS.

4.4.1. State representation

At decision step t , the state vector is

$$S_t = \left(\frac{t}{T}, \frac{Q_t}{Q_0}, P_t, r_t^{(1)}, r_t^{(2)} \right).$$

The first two entries give normalized time and remaining inventory. The third entry is the current midprice. The final two entries are lagged returns:

$$r_t^{(1)} = \frac{P_t}{P_{t-1}} - 1, \quad r_t^{(2)} = \frac{P_{t-1}}{P_{t-2}} - 1. \quad (24)$$

No order-book, volume, trade-flow, or cross-asset variables are included. The state records the execution clock, remaining inventory, and recent price movement.

4.4.2. Predictive content of the price features

Before training the policies, the lagged returns were assessed for directional information. The next-step return r_{t+1} was regressed on $(r_t^{(1)}, r_t^{(2)})$ over the intraday sample. The estimated R^2 is close to zero, and the coefficients are statistically

indistinguishable from zero. The lagged returns therefore provide recent price context, but little evidence of useful short-horizon predictability.

4.5. Model parameters and calibration

Each trading day is divided into $T = 8$ execution intervals, giving the policy several opportunities to adjust the schedule while keeping the horizon short.

The parent order is fixed at $Q_0 = 10,000$ shares. Since IS and rewards are normalized by Q_0 , the main comparison is about execution timing rather than order size.

Transaction costs consist of a constant half-spread $s = 0.01$ and linear temporary impact. The impact term is proportional to q_t/Q_0 , with coefficient $k = 0.05$, penalizing concentrated trades without requiring a separate impact-calibration exercise.

Execution prices also include bounded residual noise,

$$\varepsilon_t \sim \text{Unif}(-\varepsilon, \varepsilon), \quad (25)$$

with $\varepsilon = 0.002$, which represents within-interval execution variation, such as queue position or short-lived changes in available liquidity. In paired comparisons, both policies receive the same realization.

The default residual bound is $\varepsilon_\delta = 0.10$, permitting moderate deviations from TWAP while keeping the learned policy close to the benchmark (Table 2).

Table 2
Model and experimental parameters

Parameter	Value
Execution horizon T	8
Initial inventory Q_0	10,000 shares
Half-spread s	0.01
Temporary impact coefficient k	0.05
Execution noise ε	0.002
Residual bound ε_δ	0.10
Discount factor γ	0.10

4.6. Sensitivity analysis of the residual bound

The residual bound determines the size of the neighborhood around TWAP. If the bound is too small, the policy has little room to change timing. If it is too large, the policy can begin to recover some of the instability seen in unconstrained runs. The policy is retrained for alternative values of ε_δ , and each version is tested on the same 1,000 out-of-sample episodes.

As shown in Table 3, the confidence intervals include zero for both values. Expanding the residual range does not produce a notable improvement over TWAP.

4.7. Paired performance evaluation

For each episode, improvement over the benchmark is defined as

$$\Delta \text{IS}_{\text{bps}} = \text{IS}_{\text{bps}}^{(\text{TWAP})} - \text{IS}_{\text{bps}}^{(\text{PPO})}. \quad (26)$$

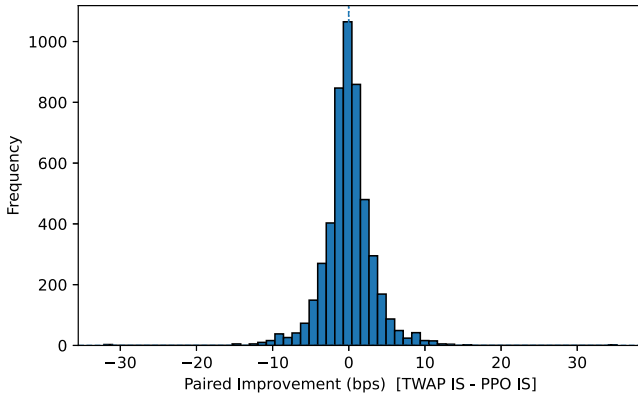
Positive values indicate that residual PPO achieves lower IS than TWAP. Since both policies trade on the same path, this is a paired pathwise statistic.

Table 3

 Sensitivity of PPO-residual performance to the residual bound ε_δ

ε_δ	Mean improvement (bps)	95% CI
0.10	-0.019	$[-0.110, 0.071]$
0.20	0.008	$[-0.081, 0.094]$

Figure 2 shows the pooled paired improvements across seeds and episodes.

 Figure 2
Distribution of paired improvements


The Wilcoxon signed-rank test gives $p = 0.57$. The null hypothesis of zero median improvement is therefore not rejected.

4.8. Multi-seed robustness

Training is repeated for five random seeds. For each seed, the mean paired improvement and a bootstrap confidence interval are estimated.

As depicted in Figure 3, the interval overlaps zero for every seed, showing that the result is not driven by a single atypical training run.

4.9. Schedule diagnostics and cost decomposition

The learned schedules are also inspected directly. Figure 4 compares TWAP with the residual policy.

Figure 3

Bootstrap confidence intervals for mean paired improvement

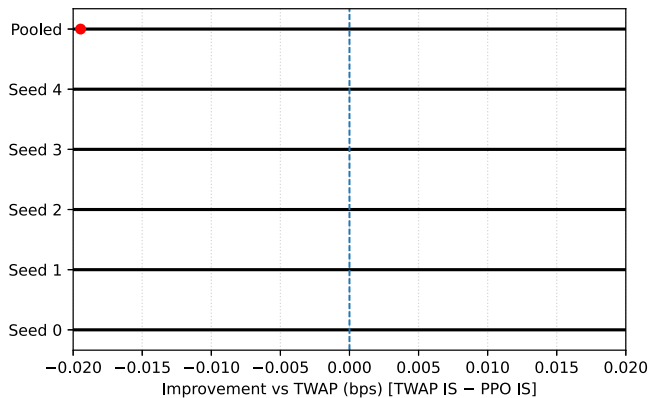
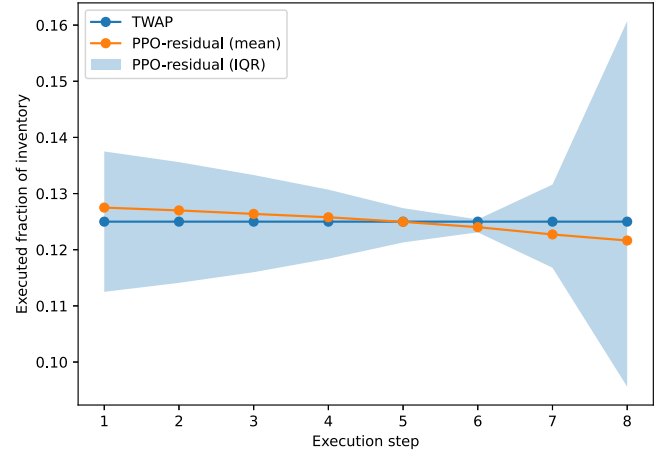


Figure 4

Execution schedule statistics for TWAP and PPO-residual policies



The residual schedules remain close to TWAP. The agent makes local timing adjustments rather than learning an entirely new liquidation curve.

The IS is decomposed into drift and friction components. Figure 5 shows that price drift explains most of the variation in outcomes. Friction costs are much more stable and are nearly identical across the two policies.

As a final check, episodes are grouped by realized volatility and drift. Within each group, the paired improvement statistic is recomputed:

$$\Delta IS_{bps} = IS_{bps}^{(TWAP)} - IS_{bps}^{(PPO)}. \quad (27)$$

On strongly trending days, the residual policy can benefit from moving volume slightly earlier or later than TWAP. The effect is not stable; however, gains in some groups are offset by losses in others. In high-volatility episodes, dispersion increases for both policies, while the average paired difference remains close to zero.

5. Discussion

The results in Section 4 highlight a difficulty in evaluating learned execution policies. IS is noisy because the trader controls the schedule, not the price path during the execution window. A policy may lower average shortfall by a few basis points without making better execution decisions. In a sell program, for example, part of the apparent gain may come from being tested on paths with more favorable intraday moves.

The dispersion of the TWAP benchmark makes this visible. Figure 1 shows that the same deterministic schedule produces a wide range of implementation shortfall outcomes on the fixed test set, including both positive and negative values. Since the schedule is unchanged across episodes, the variation must come from the realized price paths. A comparison with TWAP is therefore meaningful only when both policies are tested on the same market realizations. Otherwise, differences in shortfall can easily be drift rather than execution skill.

Some PPO and value-based Double Deep Q-Network (DDQN) runs look favorable if they are judged only by average performance. Under paired pathwise evaluation, the gains shrink and are not statistically robust. The schedules also vary

substantially across training seeds. That instability is consistent with the information available to the agent: the action space is flexible, but the state variables provide little evidence about the next price movement.

Looking across multiple seeds helps clarify whether the result is stable. One favorable training run is not strong evidence of a reliable execution rule. Initialization, exploration noise, and stochastic gradient updates can all affect the final policy. A positive result from a single seed may reflect the training sample, the optimization path, or temporary execution noise. Pairing the price paths, and also the execution-noise realizations when noise is present, removes much of this ambiguity.

The fixed out-of-sample paths are used only for evaluation, and to make the comparison fair, on each episode, TWAP and the learned policy face the same realized market path. Without that condition, the assignment of test episodes to policies could itself create differences in IS.

The agent observes the execution clock, remaining inventory, current price information, and two lagged returns. It does not observe order-book imbalance, trade flow, liquidity shocks, volume forecasts, or cross-asset variables. With such information, RL might have a stronger basis for adaptation. Cheng et al. [24], for instance, discuss settings with high-dimensional states and strategic interaction, where adaptive methods are more naturally motivated. The experiment here is narrower: one asset, one agent, explicit costs, and weak short-horizon predictors. Under these conditions, the learned policy does not reliably outperform TWAP.

The residual formulation does change the learned schedules. By forcing the policy to stay close to TWAP, it removes the most extreme front-loaded and back-loaded patterns seen in the unconstrained runs. The resulting schedules are smoother and less sensitive to the training seed, as Figure 4 shows.

In the largest evaluation, using 1,000 fixed episodes and multiple seeds, paired improvements are close to zero (Figure 2). The confidence intervals in Figure 3 include zero for every seed. As a result, the residual policy should be interpreted as a bounded perturbation of TWAP, not as a policy that systematically lowers IS.

The drift–friction decomposition in Figure 5 shows that most of the variation in IS comes from the drift component. The friction component is more stable and very similar across policies. For a sell order, the sign of IS can also be misleading: upward

drift can produce negative shortfall even when the timing decision is unremarkable, while downward drift can produce positive shortfall under a reasonable schedule. The more relevant question is whether the learned policy reduces frictional costs on the same path. In these experiments, it does not.

The simulator uses a fixed half-spread, linear temporary impact, and bounded residual execution noise, leaving out several realistic features of limit-order-book dynamics, such as concave impact, transient impact decay, and spreads that change with volatility or liquidity. The simplification keeps the main cost channels visible and makes the drift–friction decomposition easier to interpret.

Changing the cost specification alone would not necessarily overturn the main result. Since both policies are evaluated on the same realized paths, alternative impact curves or state-dependent spreads would still be applied to the same episodes. Persistent outperformance would require the policy to exploit information about future prices, liquidity, or impact. The state variables used here do not appear to contain that information.

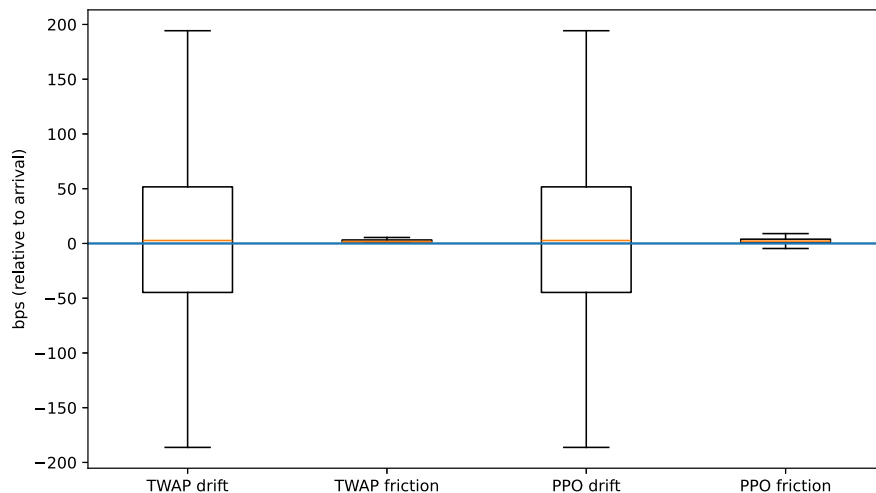
The preliminary predictability check in Section 4 supports the interpretation. The two lagged returns explain none of the next-step return variation essentially. The negative result should therefore be read mainly as an information result, not as a claim that PPO, or RL more generally, is unsuitable for execution.

Sparse-state information, explicit frictions, and substantial intraday drift leave little room for adaptive timing to improve mean shortfall. More realistic simulators are still important, particularly those with concave impact, transient impact, and state-dependent liquidity. But a richer cost model does not by itself create predictability. A more promising direction is to combine richer microstructure dynamics with state variables that actually contain information about liquidity, order flow, or the state of the book.

Finally, the residual policy may still be useful. Execution algorithms are not judged only by average shortfall. Stability, interpretability, controllability, and behavior under stress also matter. A residual architecture satisfies several of these requirements: it keeps the learned policy close to a known benchmark, limits pathological schedules, and makes deviations from the baseline easy to inspect.

The role of RL in this setting is better viewed as a constrained adjustment mechanism than as a replacement for time-based execution. It may be useful for controlled adaptation and diagnostics,

Figure 5
Decomposition of implementation shortfall into drift and friction components



especially for identifying regimes in which a benchmark schedule behaves poorly. The evidence here does not support persistent average outperformance when the information available to the agent is weak.

6. Conclusions

This paper compared RL execution policies with TWAP using an evaluation design described above. The policies were tested on fixed out-of-sample intraday paths, compared through paired episode-level differences, trained across multiple random seeds, and evaluated with explicit transaction costs. The goal was to separate execution effects from variation due to intraday drift, stochastic training, and the particular composition of the test set.

The unconstrained agents perform poorly under the protocol. When the policy can choose execution quantities freely, the learned schedules are unstable and do not reliably improve on TWAP once costs are included. Some individual runs produce favorable average shortfall, but these gains disappear after paired evaluation and aggregation across seeds, matching the information structure of the experiment: the state contains little short-horizon predictive content, while realized shortfall is strongly affected by price movements outside the agent's control.

The residual formulation addresses the stability issue; the performance issue remains open. By restricting the policy to bounded deviations from TWAP, it removes the most extreme front-loaded and back-loaded schedules. The resulting policies are smoother, less sensitive to the training seed, and easier to inspect. In the out-of-sample tests, residual RL remains close to TWAP and does not deliver a statistically reliable reduction in IS.

The drift–friction decomposition explains the result. Most of the variation in IS comes from the realized intraday price path rather than from differences in frictional costs. On the same paths, TWAP and the residual policy incur very similar frictions. Without informative signals about future prices, liquidity, or impact, the agent has little basis for producing persistent value through timing.

The results do not show that RL is unsuitable for execution in general. They show that in this setting—one asset, sparse-state information, explicit transaction costs, and paired out-of-sample evaluation—RL is better understood as a constrained adjustment mechanism, as its value lies essentially in stabilizing behavior, limiting pathological schedules, and making deviations from TWAP easier to diagnose.

Future work will include order-book variables, liquidity measures, trade-flow indicators, cross-asset signals, and state-dependent impact. Risk-sensitive objectives are also important, since execution desks may care more about avoiding adverse tail outcomes than about small improvements in mean shortfall. Multi-agent or market-impact-consistent simulators could further clarify whether learning becomes more useful when execution decisions interact with other market participants.

Funding Support

This work is supported by Linea di intervento D.1 – A Survey about Green AI.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The data that support the findings of this study are openly available in GitHub at <https://github.com/EBarbiero/Reassessing-Reinforcement-Learning-for-Optimal-Trade-Execution>.

Author Contribution Statement

Matteo Tosi: Conceptualization, Methodology, Software, Investigation, Resources, Data curation, Visualization. **Enrico Barbierato:** Conceptualization, Software, Formal analysis, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Alice Gatti:** Validation, Writing – review & editing.

References

- [1] Donnelly, R. (2022). Optimal execution: A review. *Applied Mathematical Finance*, 29(3), 181–212. <https://doi.org/10.1080/1350486X.2022.2161588>
- [2] Cartea, Á., Jaimungal, S., & Penalva, J. (2015). *Algorithmic and high-frequency trading*. Cambridge University Press.
- [3] Fruth, A., Schöneborn, T., & Urusov, M. (2019). Optimal trade execution in order books with stochastic liquidity. *Mathematical Finance*, 29(2), 507–541. <https://doi.org/10.1111/mafi.12180>
- [4] Ivanov, I. O. (2023). The efficient market hypothesis: Scientific consensus on the viability of the theory. *Modern Economics*, 39(1), 46–53. [https://doi.org/10.31521/modecon.V39\(2023\)-07](https://doi.org/10.31521/modecon.V39(2023)-07)
- [5] Kim, S., Kim, J., Sul, H. K., & Hong, Y. (2024). An adaptive dual-level reinforcement learning approach for optimal trade execution. *Expert Systems with Applications*, 252, 124263. <https://doi.org/10.1016/j.eswa.2024.124263>
- [6] Tonkin, I., Gepp, A., Harris, G., & Vanstone, B. J. (2025). Benchmarking deep reinforcement learning approaches to trade execution. *Pacific-Basin Finance Journal*, 94, 102876. <https://doi.org/10.1016/j.pacfin.2025.102876>
- [7] Byun, W. J., Choi, B., Kim, S., & Jo, J. (2023). Practical application of deep reinforcement learning to optimal trade execution. *FinTech*, 2(3), 414–429. <https://doi.org/10.3390/fintech2030023>
- [8] Nagy, P., Calliess, J.-P., & Zohren, S. (2023). Asynchronous deep double dueling Q-learning for trading-signal execution in limit order book markets. *Frontiers in Artificial Intelligence*, 6, 1151003. <https://doi.org/10.3389/frai.2023.1151003>
- [9] Cont, R., Degond, P., & Xuan, L. (2025). A mathematical framework for modeling order book dynamics. *SIAM Journal on Financial Mathematics*, 16(1), 123–166. <https://doi.org/10.1137/22M1541538>
- [10] Arian, H., Norouzi Mobarekeh, D., & Seco, L. (2024). Back-test overfitting in the machine learning era: A comparison of out-of-sample testing methods in a synthetic controlled environment. *Knowledge-Based Systems*, 305, 112477. <https://doi.org/10.1016/j.knsys.2024.112477>
- [11] Horst, U., & Kivman, E. (2024). Optimal trade execution under small market impact and portfolio liquidation with semimartingale strategies. *Finance and Stochastics*, 28(3), 759–812. <https://doi.org/10.1007/s00780-024-00536-2>

- [12] Macri, A., & Lillo, F. (2024). Reinforcement learning for optimal execution when liquidity is time-varying. *Applied Mathematical Finance*, 31(5), 312–342. <https://doi.org/10.1080/1350486X.2025.2490157>
- [13] Annaswamy, A. M. (2023). Adaptive control and intersections with reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 6(1), 65–93. <https://doi.org/10.1146/annurev-control-062922-090153>
- [14] Meng, W., Zheng, Q., Pan, G., & Yin, Y. (2023). Off-policy proximal policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(8), 9162–9170. <https://doi.org/10.1609/aaai.v37i8.26099>
- [15] Dai, J., Ji, J., Yang, L., Zheng, Q., & Pan, G. (2023). Augmented proximal policy optimization for safe reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6), 7288–7295. <https://doi.org/10.1609/aaai.v37i6.25888>
- [16] Sadoghi, A., & Vecer, J. (2022). Optimal liquidation problem in illiquid markets. *European Journal of Operational Research*, 296(3), 1050–1066. <https://doi.org/10.1016/j.ejor.2021.05.020>
- [17] Maglaras, C., Moallemi, C. C., & Wang, M. (2022). A deep learning approach to estimating fill probabilities in a limit order book. *Quantitative Finance*, 22(11), 1989–2003. <https://doi.org/10.1080/14697688.2022.2124189>
- [18] Roch, A. (2023). Optimal liquidation through a limit order book: A neural network and simulation approach. *Methodology and Computing in Applied Probability*, 25(1), 3. <https://doi.org/10.1007/s11009-023-09996-z>
- [19] Fu, G., Hager, P. P., & Horst, U. (2024). Mean-field liquidation games with market drop-out. *Mathematical Finance*, 34(4), 1123–1166. <https://doi.org/10.1111/mafi.12429>
- [20] Cordoni, F., & Lillo, F. (2024). Transient impact from the Nash equilibrium of a permanent market impact game. *Dynamic Games and Applications*, 14(2), 333–361. <https://doi.org/10.1007/s13235-023-00497-9>
- [21] Schnaubelt, M. (2022). Deep reinforcement learning for the optimal placement of cryptocurrency limit orders. *European Journal of Operational Research*, 296(3), 993–1006. <https://doi.org/10.1016/j.ejor.2021.04.050>
- [22] Buehler, H., Gonon, L., Teichmann, J., & Wood, B. (2019). Deep hedging. *Quantitative Finance*, 19(8), 1271–1291. <https://doi.org/10.1080/14697688.2019.1571683>
- [23] Mascioli, C., Gu, A., Wang, Y., Chakraborty, M., & Wellman, M. (2024). A financial market simulation environment for trading agents using deep reinforcement learning. In *Proceedings of the 5th ACM International Conference on AI in Finance*, 117–125. <https://doi.org/10.1145/3677052.3698639>
- [24] Cheng, L., Wei, X., Li, M., Tan, C., Yin, M., Shen, T., & Zou, T. (2024). Integrating evolutionary game-theoretical methods and deep reinforcement learning for adaptive strategy optimization in user-side electricity markets: A comprehensive review. *Mathematics*, 12(20), 3241. <https://doi.org/10.3390/math12203241>
- [25] Zhang, C., Duan, Y., Chen, X., Chen, J., Li, J., & Zhao, L. (2023). Towards generalizable reinforcement learning for trade execution. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 4975–4983. <https://doi.org/10.24963/ijcai.2023/553>

How to Cite: Tosi, M., Barbierato, E., & Gatti, A. (2026). Reassessing Reinforcement Learning for Optimal Trade Execution. *Journal of Computational and Cognitive Engineering*. <https://doi.org/10.47852/bonviewJCCE62028887>