Multi-Trajectory Physics-Informed Neural Networks for HJB Equations with Hard-Zero Terminal Inventory: Optimal Execution on Synthetic & SPY Data

Anthime Valin*
Department of Computer Science
University College London
anthime.valin@gmail.com

Abstract

We study optimal trade execution with a hard-zero terminal inventory constraint, modeled via Hamilton-Jacobi-Bellman (HJB) equations. Vanilla PINNs often under-enforce this constraint and produce unstable controls. We propose a Multi-Trajectory PINN (MT-PINN) that adds a rollout-based trajectory loss and propagates a terminal penalty on X_T via backpropagation-through-time, directly enforcing $X_T = 0$. A lightweight λ-curriculum is adopted to stabilize training as the state expands from a risk-neutral reduced HJB to a risk-averse HJB. On the Gatheral-Schied single-asset model [1], MT-PINN aligns closely with their derived closed-form solutions and concentrates terminal inventory tightly around zero while reducing errors along optimal paths. We apply MT-PINNs on SPY intraday data, matching TWAP when risk-neutral, and achieving lower exposure and competitive costs, especially in falling windows, for higher risk-aversion.

1 Introduction

Many financial decision-making problems, ranging from optimal portfolio and consumption choice to optimal order execution, can be written as optimal control problems, with the value function governed by a Hamilton-Jacobi-Bellman (HJB) PDE [2]. In optimal trade execution, a necessary constraint is hard-zero terminal inventory (i.e, all shares must be liquidated by the end of the horizon). This requirement creates a singularity near terminal ($\tau = T - t \to 0$), where the value function exhibits non-smooth structure. Vanilla PINNs, trained on PDE residuals and soft boundary penalties, frequently under-enforce this constraint [3], often yielding non-zero inventory in simulated rollouts and unstable control near maturity.

We propose a Multi-Trajectory PINN (MT-PINN) that directly penalizes terminal inventory via a rollout-based trajectory loss, using backpropagation-through-time (BPTT) to propagate the terminal penalty through simulated trajectories (see Appendix B). This decreases terminal inventory violations and reduces error along optimal control paths. A lightweight λ-curriculum (from risk-neutral to risk-averse) is employed to further stabilize training. We evaluate MT-PINNs' performance on a single-asset execution model, originally proposed by Gatheral and Schied in [1], demonstrating its capabilities in aligning with the closed-form solution trading rates and concentrating terminal inventory tightly around zero compared to vanilla PINNs. Furthermore, we apply MT-PINNs to SPY intraday data, showing risk aversion explicitly reflected in the expected exposure-cost trade-off,

*This work was conducted in partial fulfillment of the requirements for the MSc in Financial Technology at University College London.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: NeurIPS 2025 Workshop on Generative AI in Finance.

coinciding with TWAP when risk neutral ($\lambda = 0$) and revealing advantages over TWAP in falling markets. Specifically, we make the following contributions: (i) a trajectory-aware loss that enforces $X_T = 0$; (ii) a simple curriculum regularization for risk aversion; and (iii) synthetic + real-market validation with strong constraint satisfaction and stable controls.

2 Problem Setup and Method

Model & notation. We study a variant of the Almgren-Chriss optimal execution model, proposed by Gatheral and Schied [1], where the unaffected price S_t follows GBM ($dS_t = \sigma S_t dW_t$, for volatility $\sigma > 0$) with market impact comprising linear permanent impact with strength $\kappa > 0$ and standard temporary impact. We work over a horizon $T > 0$ with time-to-maturity $\tau = T - t$ and initial inventory $X_0 > 0$. Inventory X_t evolves under the sell trading rate v_t via $\dot{X}_t = -v_t$ with a hard terminal constraint $X_T = 0$. The value function is denoted by $\Gamma(\tau, X, S)$ in the risk-averse case and $\Gamma(\tau, X)$ in the risk-neutral case. It represents the minimum expected cost from the current state to the terminal, accounting for market impact and, when applicable, risk aversion λ.

Reduced HJB. The HJB equation takes different forms in the risk-neutral and risk-averse regimes (the risk-neutral case is S-invariant). Minimizing the Hamiltonian with respect to v and substituting the optimal feedback $v^* = \frac{1}{2}\partial\Gamma/\partial X$, yields the reduced HJB:

$$\frac{\partial \Gamma}{\partial \tau} = \begin{cases} \kappa^2 X^2 - \dfrac{1}{4}\Big(\dfrac{\partial \Gamma}{\partial X}\Big)^2, & \lambda = 0, \ \text{(Proposition A.1)} \\ \dfrac{1}{2}\sigma^2 S^2 \dfrac{\partial^2 \Gamma}{\partial S^2} + \kappa^2 X^2 + \lambda S X - \dfrac{1}{4}\Big(\dfrac{\partial \Gamma}{\partial X}\Big)^2, & \lambda > 0 \ \text{(Proposition A.2).} \end{cases}$$

with terminal condition $\Gamma(\tau \to 0,\, X, S) = 0$ if $X = 0$ and $+\infty$ otherwise. At $\lambda = 0$ the value is price-invariant, so the state dimension is 1 (variables (τ, X)), while for $\lambda > 0$ it is 2 (variables (τ, X, S)).

MT-PINN parameterization. We approximate the value function with a smooth MLP $\hat{\Gamma}(\tau, X, S; \theta)$ (independent of S when $\lambda = 0$). Derivatives found in the HJB equation are calculated using autodiff and the feedback control used for rollouts is $v^* = \frac{1}{2}\,\partial\hat{\Gamma}/\partial X$.

Multi-trajectory terminal-inventory loss. To enforce the hard constraint $X_T = 0$, we add a rollout-based loss that penalizes terminal inventory across a batch of starting states and horizons. We fix the sets of initial inventories and prices $\{(X_0^{(p)}, S_0^{(p)})\}_{p=1}^{P} \subset \mathcal{X}_0 \times \mathcal{S}_0$ and horizons $\{T_j\}_{j=1}^{J} \subset (0, T]$. For each pair (p, j), define $x_{T_j}^{(p)}$ as the terminal inventory obtained by rolling out the inventory dynamics under the network-implied control, using a forward Euler discretization with N_{dt} steps:

$$X_{k+1} = X_k - v^*(\tau_k, X_k, S_k)\Delta t, \ \ \tau_k = T_j - t_k, \ \ \Delta t = \frac{T_j}{N_{\text{dt}}}.$$

The loss averages a composite penalty, ψ, across the batch:

$$\mathcal{L}_{\text{traj}} = \frac{1}{PJ}\sum_{p=1}^{P}\sum_{j=1}^{J} \psi(x_{T_j}^{(p)}), \quad \psi(x_T) = \begin{cases} |x_T|, & |x_T| \le 1, \\ x_T^2, & |x_T| > 1. \end{cases}$$

For $\lambda = 0$ the rollouts depend only on $\Big(X_0^{(p)}, T_j\Big)$; for $\lambda > 0$ they depend on $\Big(X_0^{(p)}, S_0^{(p)}, T_j\Big)$. Gradients are computed via backpropagation-through-time (BPTT) (see Appendix B).

Composite loss. We then minimize a composite loss:

$$\mathcal{L}_{\text{total}}(\theta; \lambda) = w_{\text{PDE}}\mathcal{L}_{\text{PDE}}(\theta; \lambda) + w_{\text{traj}}\mathcal{L}_{\text{traj}}(\theta; \lambda) + w_{\text{IC}}\mathcal{L}_{\text{IC}}(\theta; \lambda) + w_{\text{sym}}\mathcal{L}_{\text{sym}}(\theta; \lambda) + \mathbf{1}_{\{\lambda>0\}} w_0 \mathcal{L}_{\text{0-term}}(\theta),$$

where $\mathcal{L}_{\text{PDE}}$ is the squared residual of the reduced HJB above, $\mathcal{L}_{\text{IC}}$ enforces the inventory-axis condition, $\mathcal{L}_{\text{sym}}$ enforces the model's symmetry (see Corollary A.2.1 and Corollary A.2.2), and $\mathcal{L}_{\text{0-term}}$ sets $\Gamma(0, 0, S) = 0$ for $\lambda > 0$. Loss weights use DWA-style adaptive weighting (see Appendix C.2).

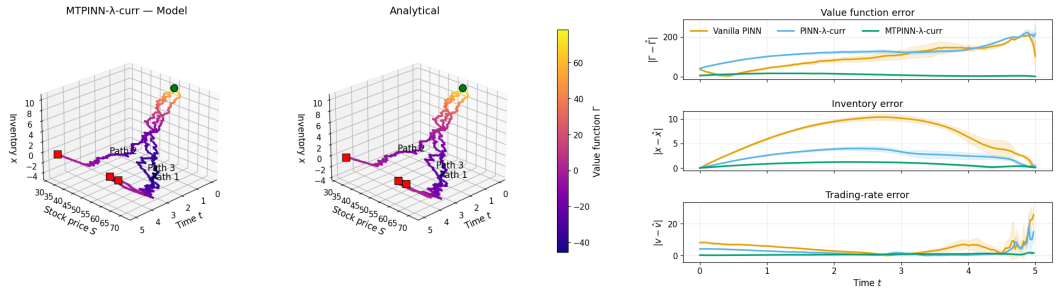


Figure 1: *Left:* MT-PINN vs analytical optimal paths for $\lambda = 0.10$ on three price paths, colored by the value function Γ with a shared colormap. *Right:* three stacked panels show mean $\pm$ one standard deviation across the same paths of absolute errors along the optimal path: (top) $|\Gamma - \hat{\Gamma}|$, (middle) $|x - \hat{x}|$, (bottom) $|v - \hat{v}|$.

Table 1: Absolute terminal-inventory enforcement on the synthetic benchmark for 200 simulated price paths and $\lambda = 0.10$ (for varying λ see Table 3 in Appendix F). We report $p_\varepsilon = \Pr(|X_T| \leq \varepsilon)$ with $\varepsilon = 0.05$.

	$\lvert X_T\rvert$ statistics		
Method	Mean $\pm$ Std	95th pct	p_ε
Vanilla PINN	0.777 ± 0.444	1.407	0.055
PINN-λ-curr	0.164 ± 0.161	0.527	0.205
MT-PINN-λ-curr	$\mathbf{0.073 \pm 0.092}$	**0.241**	**0.600**

λ-curriculum. Following [4], we apply a curriculum regularization to the risk-aversion parameter λ. We choose a monotone schedule $0 = \lambda_0 < \lambda_1 < \cdots < \lambda_k = \lambda^*$ and, at stage k, minimize $\mathcal{L}_{\text{total}}(\theta; \lambda_k)$, warm-starting from the previous stage ($\theta^{(k)} \leftarrow \theta^{k-1}$). Training begins at $\lambda_0 = 0$, where the value function is price-invariant, and then transitions to $\lambda > 0$. This curriculum has been shown to stabilize training and yield lower PDE residuals [4, 5].

3 Synthetic Benchmark

Baselines. We compare MT-PINN against two vanilla PINNs that omit the multi-trajectory term and instead use a quadratic terminal penalty [2]: (i) Vanilla PINN, trained with HJB residual, terminal value penalty, and same structural terms as MT-PINN); (ii) Vanilla PINN + λ-curriculum, with the same loss as (i), but trained with the staged λ schedule as defined in §2.

Setup. Fixed horizon $T = 5.0$; inventory domain $X \in [-10, 10]$ and $S \in [10, 100]$; Volatility $\sigma = 0.1$; permanent-impact strength $\kappa = 0.1$; Collocation points per run $N_{\text{PDE}} = 30,000$; For $\lambda > 0$, baselines and MT-PINN trained on 55k epochs; target risk aversion $\lambda^* \in \{0, 0.05, 0.1\}$ with five curriculum stages with $\lambda_\alpha = \alpha \cdot \lambda^*$, for $\alpha \in (0.25, 0.50, 0.75, 0.9, 1.0)$ (used for MT-PINN and baseline (ii)). The multi-trajectory batch samples $P = 820$ initial state $\{(X_0^{(0)}, S_0^{(p)})\}_{p=1}^P$ (omitting $S_0^{(p)}$ when $\lambda = 0$) and rolls out to horizons $\{T/50, T/10, T/5, 2T/5, 3T/5, 4T/5, T\}$ with 200 Euler steps. All remaining details are provided in Appendix D.

Key Results. MT-PINN attains the lowest error along the optimal path across absolute value function error, inventory error, and trading rate error, clearly outperforming both baselines over most of the horizon and closely tracking the analytical control without the late-maturity instability seen in the baseline PINNs (see Figure 1). This translates into tighter enforcement of the hard-zero terminal constraint as evidenced in Table 1. Additional metrics, residual maps, and results comparing varying $\lambda \in [0, 0.05, 0.10]$ are provided in Appendix F.

[2]Empirically, it was found that if neither multi-trajectory nor this quadratic terminal penalty is used, the PINNs will converge to sub-optimal local minima.

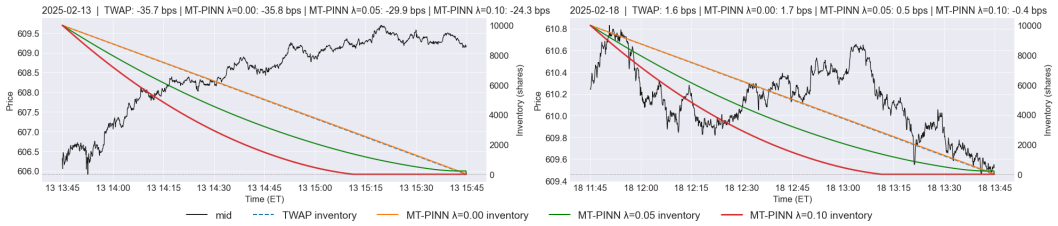


Figure 2: Execution trajectories and costs for MT-PINN with varying $\lambda \in [0, 0.05, 0.1]$ versus TWAP, over two sampled trading windows of the SPY. *Left:* rising SPY window; *Right:* falling SPY window.

Table 2: Execution exposure and cost comparison of MT -PINN (varying λ) vs. TWAP. Results averaged over $n = 21$ windows. Costs in basis points (1 bps $= 0.01\%$).

Model	**Mean Exposure**	**Exposure Std.**	**Mean Cost (bps)**	**Cost Std. (bps)**
TWAP	0.334	0.0000	-6.35	12.56
MT-PINN $\lambda = 0.00$	0.336	0.0000	-6.37	12.58
MT-PINN $\lambda = 0.05$	0.231	0.0004	-5.01	11.02
MT-PINN $\lambda = 0.10$	0.164	0.0002	-3.69	9.67

4 Real-Market SPY Backtest

Setup. The trading day on SPY mid-price is split into three 2-hour fixed intraday windows on 5-second intervals, and repeated across seven days (Feb 10 - Feb 19, 2025, excluding weekends)[3], and assumes no trading fees or overnight risk. Uses the same Gatheral-Schied dynamics/penalties as §2. MT-PINN with $\lambda \in \{0, 0.05, 0.10\}$ is compared against TWAP; $\lambda = 0$ serves as the risk-neutral MT-PINN which should coincide with TWAP in expectation. Time horizon is expressed by trading day units, where elapsed time is normalized by the effective length of a trading day on the U.S. equity markets (i.e., $T = 0.308$). Inventory is normalized by the initial position, so that the inventory lies in the interval $[-1, 1]$, and represents the fraction of the initial inventory. Stock price $S \in [590, 620]$ is set based on the mid-price range over the full 21 time windows. Daily realized volatility is calculated from the data and is set to $\sigma = 0.0038$ ($\approx 6\%$ annualized), and permanent-impact strength is set to $\kappa = 0.2$. The MT-PINN setup follows similarly to §3. For more details see Appendix E.

Key Results. MT-PINN with $\lambda = 0$ closely matches TWAP, both in exposure (0.336 vs. 0.334) and cost (-6.37 vs. -6.35 bps), confirming the risk-neutral case (see Table 2). Increasing risk aversion traces a clean risk-cost frontier: mean exposure drops while average cost rises moderately (likely due to the general uptrend of the SPY). Figure 2 shows $\lambda > 0$ front-loads the execution and outperforms TWAP in down-moves, whereas MT-PINN (with $\lambda = 0$)/TWAP remains competitive in rising windows. Across all windows, MT-PINN policies remain smooth and strongly satisfy the zero terminal inventory constraint, aligning with the synthetic benchmark (see Appendix F for more results).

Reproducibility. Code, configs, and scripts to reproduce all figures and experiments are provided at: `https://github.com/anthimevalin/Multi-Trajectory-PINNs-Zero-Terminal-HJB`.

5 Conclusion & Limitations

MT-PINN enforces hard-zero terminal inventory via a rollout-based trajectory loss and, with a simple λ-curriculum, produces stable controls and less error along the optimal path. On the Gatheral–Schied benchmark it matches the closed-form control and concentrates X_T tightly around zero; on SPY intraday data, $\lambda = 0$ behaves like TWAP, while $\lambda > 0$ traces a clear risk–cost frontier and performs best in falling windows.

[3]Windows exclude the first 15 minutes of market open and the last 15 minutes of market close. This was done to avoid the noisy and elevated volatility typically observed at those times [6].

Limitations. The impact model used doesn't include fees, nonlinear temporary impact, and assumes frictionless execution beyond impact terms; experiments are single-asset and don't investigate cross-asset; and the trajectory penalty has per-epoch time complexity $\mathcal{O}(J \times N_{\text{dt}} \times P)$ and memory complexity $\mathcal{O}(N_{\text{dt}} \times P)$ (more details in Appendix B). Moreover, the execution model abstracts away market microstructure: execution occurs in a continuous-price setting without modeling order book depth, bid–ask dynamics, or order-flow imbalance. Consequently, the learned policies may overlook short-term liquidity constraints and queue dynamics that affect realized execution costs. Incorporating order-book-derived state variables would be a natural extension toward more realistic liquidity modeling.

References

[1] Jim Gatheral and Alexander Schied. Optimal trade execution under geometric brownian motion in the almgren and chriss framework. *International Journal of Theoretical and Applied Finance*, 14(03):353–368, 2011.

[2] Huyên Pham. *Continuous-time stochastic control and optimization with financial applications*, volume 61. Springer Science & Business Media, 2009.

[3] Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*, 2022.

[4] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in neural information processing systems*, 34:26548–26560, 2021.

[5] Yao Huang, Wenrui Hao, and Guang Lin. Hompinns: Homotopy physics-informed neural networks for learning multiple solutions of nonlinear elliptic differential equations. *Computers & Mathematics with Applications*, 121:62–73, 2022.

[6] A. Can Inci and Deniz Ozenbas. Intraday volatility and the implementation of a closing call auction at borsa istanbul. *Emerging Markets Review*, 33:79–89, 2017. ISSN 1566-0141. doi: https://doi.org/10.1016/j.ememar.2017.09.002. URL `https://www.sciencedirect.com/science/article/pii/S1566014117303552`.

[7] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019.

A Mathematical details & proofs

We seek the optimal strategy to liquidate an initial inventory of $X > 0$ shares over some fixed time horizon $[0, T]$. The unaffected asset price follows a driftless geometric Brownian motion (GBM):

$$dS_t = \sigma S_t dW_t, \quad S_0 > 0$$

Gatheral and Schied in [1] assume that the number of shares is represented by an absolutely continuous adapted process X_t that evolves as:

$$X_t = X - \int_0^t v_s ds,$$

with boundary conditions $X_0 = X$ and $X_T = 0$ and where $v_t = \dot{X}_t \in \mathbb{R}$ is the trading rate at time t. Furthermore, transactions occur at

$$\tilde{S}_t = S_t + \eta v_t + \gamma(X_t - X_0),$$

where S_t is the unaffected stock price process, ηv_t models temporary impact, and $\gamma(X_t - X_0)$ models permanent impact. Following [1], the temporary impact coefficient is scaled to one and the effect of permanent impact is represented by a quadratic inventory penalty with parameter κ.

The control problem minimizes the expected cumulative cost

$$J = \mathbb{E}\left[\int_0^T (v_t^2 + \kappa^2 X_t^2 + \lambda X_t S_t) dt\right],$$

where $\kappa > 0$ weights inventory-holding, $\lambda \geq 0$ denotes the risk aversion parameter (VaR-based penalty on inventory exposure), and v_t^2 encodes the temporary impact costs.

We formulate two risk-aversion regimes in the HJB equation: when $\lambda = 0$ and when $\lambda > 0$.

Proposition A.1 (Risk-Neutral HJB)**.** *For zero risk penalty (i.e., $\lambda = 0$), the resulting stochastic control problem, whose value function $\Gamma(\tau, X)$ with $\tau := T - t$ satisfies the nonlinear HJB equation under optimal control*

$$\frac{\partial \Gamma}{\partial \tau} = \kappa^2 X^2 - \frac{1}{4}\left(\frac{\partial \Gamma}{\partial X}\right)^2 \tag{1}$$

with

$$\Gamma(\tau, 0) = 0, \ \ \forall \tau \in [0, T] \tag{2}$$

subject to the terminal condition

$$\Gamma(0, X) = \begin{cases} 0, & \text{if } X = 0 \\ +\infty, & \text{otherwise.} \end{cases} \tag{3}$$

Proof. The cost functional for zero risk penalty is

$$J(\tau, X) = \mathbb{E}\left[\int_0^\tau (v_t^2 + \kappa^2 X_t^2) dt\right],$$

and thereby no longer dependent on the price S_t. The value function can then be written as the infimum of the cost functional

$$\Gamma(\tau, X) = \inf_{v \in \mathbb{R}} \mathbb{E}\left[\int_0^\tau ((v_t^2 + \kappa^2 X_t^2) dt)\right],$$

Using the Principle of Optimality, we can represent the value function as:

$$\Gamma(\tau, X) = \inf_{v_{[t,t+\Delta t]}} \int_t^{t+\Delta t} (v_s^2 + \kappa^2 X_s^2) ds + \Gamma(\tau - \Delta t, X(t + \Delta t)) \tag{4}$$

For small Δt:

$$\int_t^{t+\Delta t} (v_s^2 + \kappa^2 X_s^2) ds = (v_t^2 + \kappa^2 X_t^2)\Delta t + \mathcal{O}(\Delta t^2), \text{ and}$$

$$\begin{aligned} X(t + \Delta t) &= X + \dot{X}\Delta t + \mathcal{O}(\Delta t^2) \\ &= X - v_t \Delta t + \mathcal{O}(\Delta t^2) \end{aligned}$$

Using Taylor expansion, the value function can be expressed by:

$$\Gamma(\tau - \Delta t, X - v_t \Delta t) = \Gamma(\tau, X) - \frac{\partial \Gamma}{\partial \tau}\Delta t - v\frac{\partial \Gamma}{\partial X}\Delta t + \mathcal{O}(\Delta t^2)$$

Inserting these back into 4 and subtracting $\Gamma(\tau, X)$ from both sides:

$$0 = \inf_{v \in \mathbb{R}} \left[v^2 + \kappa^2 X^2 - \frac{\partial \Gamma}{\partial \tau} - v\frac{\partial \Gamma}{\partial X} \right] \Delta t + \mathcal{O}(\Delta t^2)$$

Dividing by Δt and letting $\Delta t \to 0$:

$$\frac{\partial \Gamma}{\partial \tau} = \kappa^2 X^2 + \inf_{v \in \mathbb{R}} \left[v^2 - v\frac{\partial \Gamma}{\partial X} \right]$$

The optimal control is calculated by minimizing the Hamiltonian $H(\frac{\partial \Gamma}{\partial X}, v) := v^2 - v\frac{\partial \Gamma}{\partial X}$ by setting its derivative with respect to v to zero. This yields the optimal trading rate with respect to the value function:

$$v^* = \frac{1}{2}\frac{\partial \Gamma}{\partial X} \tag{5}$$

This further reduces the HJB to obtain (1)

$$\frac{\partial \Gamma}{\partial \tau} = \kappa^2 X^2 - \frac{1}{4}\left(\frac{\partial \Gamma}{\partial X}\right)^2$$

Note that the cost functional is minimized when implementing the "do nothing" control (i.e., $X_t = 0$), which results in the condition 2:

$$\Gamma(\tau, 0) = 0, \quad \forall \tau \in [0, T]$$

□

Proposition A.2 (Risk-Averse HJB)**.** *For positive risk penalty (i.e., $\lambda > 0$), the resulting stochastic control problem, whose value function $\Gamma(\tau, X, S)$ with $\tau := T - t$ satisfies the nonlinear HJB equation under optimal control*

$$\frac{\partial \Gamma}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 \Gamma}{\partial S^2} + \kappa^2 X^2 + \lambda S X - \frac{1}{4}\left(\frac{\partial \Gamma}{\partial X}\right)^2, \tag{6}$$

with

$$\Gamma(\tau, 0, S) \leq 0 \quad \forall \tau \in [0, T], \ \forall S > 0, \tag{7}$$

subject to terminal condition

$$\Gamma(0, X, S) = \begin{cases} 0, & \text{if } X = 0 \\ +\infty, & \text{otherwise} \end{cases} \tag{8}$$

Proof. Gatheral and Schied in [1] derive the HJB equation in equation (3.18) as

$$\frac{\partial \Gamma}{\partial \tau} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 \Gamma}{\partial S^2} + \kappa^2 X^2 + \lambda S X + \inf_{v \in \mathbb{R}} \left(v^2 - v \frac{\partial \Gamma}{\partial X} \right), \tag{9}$$

Reducing the HJB using 5, we obtain 6. To obtain 7, suppose a deterministic $y \in H_0^1(0,\tau)$ with non-zero integral and such that $y(0) = y(\tau) = 0$, e.g., $y(t) = \sin(\frac{\pi t}{\tau})$. For $\epsilon > 0$ set:

$$X_t^{(\epsilon)} := \epsilon y(t), \quad v_t^{(\epsilon)} := \dot{X}_t^{(\epsilon)} = \epsilon \dot{y}(t)$$

As $X_0^{(\epsilon)} = X_\tau^{(\epsilon)} = 0$, $X^{(\epsilon)} \in \mathbb{R}$. Its cost functional is

$$J(\tau, X, S) = \mathbb{E}\left[\int_0^\tau \left(\left(v_t^{(\epsilon)} \right)^2 + \kappa^2 \left(X_t^{(\epsilon)} \right)^2 + \lambda X_t^{(\epsilon)} S_t \right) dt \right]$$

As $X^{(\epsilon)}$ is deterministic and $\mathbb{E}[S_t] = S_0$ for driftless GBM:

$$\mathbb{E}\left[\int_0^\tau \lambda X_t^{(\epsilon)} S_t dt \right] = \lambda S_0 \epsilon \int_0^\tau y(t) dt,$$

such that

$$J(\tau, X, S) = \underbrace{\epsilon^2 \int_0^\tau \dot{y}(t)^2 + \kappa^2 y(t)^2 dt}_{:= A\epsilon^2} + \underbrace{\lambda S_0 \epsilon \int_0^\tau y(t) dt}_{:= B\epsilon}$$

From Proposition A.1, we know $A\epsilon^2 \geq 0$. Choose sign of y so that $B \leq 0$. Then for sufficiently small $\epsilon \in [0, -B/2A]$:

$$J(\tau, X, S) = A\epsilon^2 + B\epsilon \leq 0, \;\; \therefore \; \Gamma(\tau, 0, S) \leq 0$$

□

Gatheral and Schied in [1] showed that the closed-form optimal strategy is:

$$X_t^* = \sinh\left((T-t)\kappa\right) \left[\frac{X}{\sinh(T\kappa)} - \frac{\lambda}{2\kappa} \int_0^t \frac{S_s}{1 + \cosh((T-s)\kappa)} ds \right]$$

with trading rate $v_t^* = -\dot{X}_t^*$, such that:

$$v_t^* = X_t^* \kappa \coth(\kappa(T-t)) + \frac{\lambda S_t}{2\kappa} \tanh\left(\frac{\kappa(T-t)}{2} \right)$$

In addition, the closed-form value function is given by:

$$\Gamma^*(\tau, X, S) = \kappa X^2 \coth(\tau\kappa) + \frac{\lambda X S}{\kappa} \tanh\left(\frac{\tau\kappa}{2}\right) - \frac{\lambda^2 S^2 e^{\sigma^2 \tau}}{4\kappa^2} \int_0^\tau \left[\tanh\left(\frac{u\kappa}{2} \right) \right]^2 e^{-\sigma^2 u} du$$

Corollary A.2.1 (Zero Risk Penalty Value Function Symmetry Property)**.** *For the value function defined in Proposition A.1 the following symmetry holds true:*

$$\Gamma(\tau, X) = \Gamma(\tau, -X) \tag{10}$$

Proof.

$$\begin{aligned} \Gamma(\tau, -X) &= \inf_{v \in \mathbb{R}} \mathbb{E}\left[\int_0^\tau v_t^2 + \kappa^2 (-X_t)^2 dt \right] \\ &= \inf_{v \in \mathbb{R}} \mathbb{E}\left[\int_0^\tau v_t^2 + \kappa^2 X_t^2 dt \right] \\ &= \Gamma(\tau, X) \end{aligned}$$

□

Corollary A.2.2 (Positive Risk Penalty Value Function Symmetry Property)**.** *For the value function defined in Proposition A.2 the following symmetry holds true:*

$$\Gamma(\tau, X, S) = \Gamma(\tau, -X, -S) \tag{11}$$

Proof.

$$\begin{aligned}\Gamma(\tau, -X, -S) &= \inf_{v\in\mathbb{R}} \mathbb{E}\left[\int_0^\tau v_t^2 + \kappa^2(-X_t)^2 + \lambda(-X_t)(-S_t)dt\right] \\ &= \inf_{v\in\mathbb{R}} \mathbb{E}\left[\int_0^\tau v_t^2 + \kappa^2 X_t^2 + \lambda X_t S_t dt\right] \\ &= \Gamma(\tau, X, S)\end{aligned}$$

□

B Core MT-PINN framework

B.1 Generic Architecture

The objective is to approximate the scalar value function $\Gamma(\tau, \xi)$ with a smooth neural network $\hat{\Gamma}(\tau, \xi; \theta)$ that supports accurate derivatives for HJB residuals. The PINNs used in this report, including MT-PINN, were developed in JAX/FLAX, using a single-head Multilayer Perceptron (MLP) with inputs $z = (\tau, \xi) \in \mathbb{R}^{1+d}$, with $\tau \in [0, T]$ (time-to-maturity) and $\xi \in \mathbb{R}^d$ (state), and with tanh smooth activation. Normalization layers and stochastic regularizers are avoided to preserve stable derivatives. Automatic differentiation was used to obtain $\partial_\tau \hat{\Gamma}$, $\nabla_\xi \hat{\Gamma}$, and second-order derivatives of the Hessian, where required, for computing the HJB residuals. Although the optimal control is not specifically parameterized, it is recovered analytically via these derivatives. MT-PINNs use the formulated optimal control derived from the HJB equation to then unroll the system dynamics along sampled trajectories and define a trajectory-based loss (viz., terminal zero-state and running-cost penalties) computed via backpropagation through time (BPTT), thereby coupling the HJB residuals with realized control performance. The intention of this approach is that the trajectory-based loss not only naturally enforces the terminal zero-state condition, but also ensures that the value function along the optimal path is accurate.

B.2 MT-PINN training mechanics (BPTT) & complexity

At each training step, trajectories, via forward Euler discretization, are simulated. At each time step the network is queried to produce the control. After reaching the terminal time, the terminal trajectory loss is computed and then Backpropagation Through Time (BPTT) is used to propagate sensitivities backward through all time steps to obtain the gradient with respect to the network parameters of that trajectory loss. Then an Adam optimizer performs one parameter update using the sum of gradients from all loss terms.

For a single trajectory, consider the loss:

$$\mathcal{L}(\theta) = \sum_{k=0}^{N-1} l_k(x_k) + l_N(x_N),$$

where l_k are all the running penalties and l_N is the terminal penalty. In many applications, a terminal-only penalty is used (i.e., $l_k \equiv 0$ for $k < N$). Nonetheless, the derivation below covers both cases.

Let θ denote all trainable network parameters and $x_k \in \mathbb{R}^n$ denote the state at step $k = 0, .., N$. Let z_k be the per-step exogenous inputs. A differentiable step map $f_\theta : \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^n$ advances the state by one time step:

$$x_{k+1} = f_\theta(x_k, z_k),$$

For an explicit forward Euler discretization of continuous dynamics $\dot{x} = g_\theta(x, z)$, we iterate the step map to obtain $x_1, ..., x_N$ and evaluate $\mathcal{L}(\theta)$ as follows:

$$x_{k+1} = x_k + g_\theta(x_k, z_k)\Delta t,$$

assuming that f_θ is differentiable in both x and θ.

Then define the sensitivity/adjoint:

$$\lambda_k := \frac{\partial \mathcal{L}}{\partial x_k} \in \mathbb{R}^n,$$

and define the local Jacobians

$$A_k := \frac{\partial f_\theta}{\partial x_k}(x_k, z_k) \in \mathbb{R}^{n\times n}, \quad B_k := \frac{\partial f_\theta}{\partial \theta}(x_k, z_k) \in \mathbb{R}^{n\times|\theta|}$$

BPTT then follows as:

$$\text{(initialization)} \quad \lambda_N = \frac{\partial l_N}{\partial x_N}(x_N),$$

$$\text{(backward recursion)} \quad \lambda_k = \frac{\partial l_k}{\partial x_k}(x_k) + A_k^\intercal \lambda_{k+1}, \;\; k = N-1, ..., 0,$$

$$\text{(parameter gradient)} \quad \nabla_\theta \mathcal{L} = \sum_{k=0}^{N-1} \left(B_k^\intercal \lambda_{k+1} + \frac{\partial l_k}{\partial \theta} \right) + \frac{\partial l_N}{\partial \theta}$$

In the case of terminal-only penalties, more specifically when $l_k \equiv 0, \text{ for } k < N \Rightarrow \partial l_k / \partial x_k = 0$, this can be simplified to:

$$\text{(backward recursion)} \quad \lambda_k = A_k^\intercal \lambda_{k+1}, \;\; k = N-1, ..., 0,$$

$$\text{(parameter gradient)} \quad \nabla_\theta \mathcal{L} = \sum_{k=0}^{N-1} \left(B_k^\intercal \lambda_{k+1} + \frac{\partial l_k}{\partial \theta} \right),$$

Practically, modern frameworks (e.g., JAX) implement this via reverse-mode autodiff, by applying vector-Jacobian products on the step function at each step to obtain updated sensitivity and parameter gradients.

Complexity. Let N_{dt} be the Euler steps per simulation, B be the number of trajectories in a batch (i.e., number of initial states $\times$ horizons sampled), and C_{f_θ} represents the cost of evaluating the network f_θ.

Time per epoch: $\mathcal{O}(N_{\text{dt}} \times B \times C_{f_\theta})$.

Space per epoch: $\mathcal{O}(N_{\text{dt}} \times B \times A)$, where A denotes the per-step activation memory per network evaluation saved for backpropagation.

Note that these complexities are equivalent to $\mathcal{O}(J \times N_{\text{dt}} \times P)$ used in the main body under $B = P \times J$. Furthermore, this new penalty is also amenable to parallelism as simulated trajectories are independent and vectorized.

Compute resources Experiments were run on 1x v6e-1 TPU and constructed using JAX/FLAX. Typical runtime ranged from 1 minute to 4 minutes.

B.3 Generic Loss Template

Let $\mathcal{C}$ denote the set of constraint/boundary conditions that are application-specific. The total loss for MT-PINNs can be then expressed as:

$$\mathcal{L}_{\text{total}} = w_{\text{PDE}}\mathcal{L}_{\text{PDE}} + w_{\text{traj}}\mathcal{L}_{\text{traj}} + \sum_{\ell \in \mathcal{C}} w_\ell \mathcal{L}_\ell,$$

$$\mathcal{L}_{\text{PDE}} = \mathbb{E}_{z\sim\mathcal{D}_{\text{PDE}}}\left[\mathcal{R}_{\text{HJB}}(z;\theta)^2\right], \;\; \mathcal{L}_{\text{traj}} = \mathbb{E}_{\zeta\sim\mathcal{D}_{\text{traj}}}\left[\psi(x_T(\zeta;\theta))\right], \;\; \psi(x_T) = \begin{cases} |x_T|, & |x_T| \le 1, \\ x_T^2, & |x_T| > 1. \end{cases}$$

where

- $\mathcal{D}_{\text{PDE}}$: is the sampling distribution of PDE collocation points over z.
- $\mathcal{D}_{\text{traj}}$: is its sampling distribution.
- $\mathcal{R}_{\text{HJB}}(z;\theta)$: is the HJB/PDE residual at z using the network.
- ζ: is a trajectory sample tuple.
- $x_T(\zeta;\theta)$: is the state at terminal time from rolling out dynamics under the control implied by the network starting at ζ.
- $\psi(x_T)$: is the terminal penalty, defined as a piecewise function to enforce zero state/inventory at maturity.
- $w_{\text{PDE}}, w_{\text{traj}}, \{w_\ell\}$ are non-negative.

See Figure 3 for an illustration of the core MT-PINN framework.

B.4 Core MT-PINN framework diagram

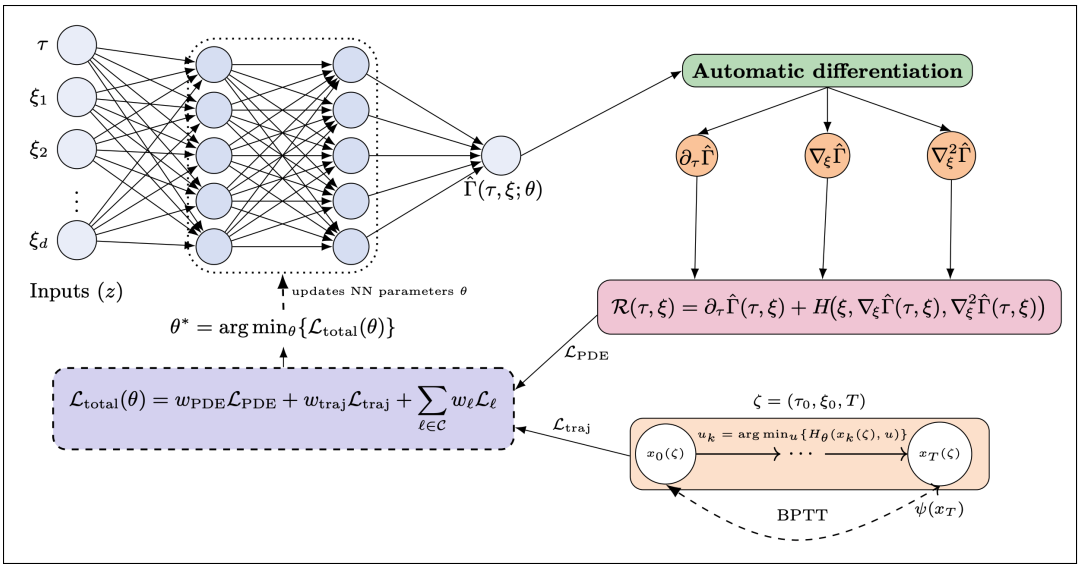


Figure 3: Core MT-PINN framework diagram. A neural network $\hat{\Gamma}(\tau,\xi;\theta)$ outputs an approximation of the value function. Automatic differentiation then provides the derivatives for the PDE. Using this, the PDE residual $\mathcal{R}(\tau,\xi)$ and the controls u_k are calculated. The PDE residual forms the PDE loss function $\mathcal{L}_{\text{PDE}}$ and the controls form the trajectory rollout penalty $\mathcal{L}_{\text{traj}}$ via BPTT. Both, contribute to the total loss $\mathcal{L}_{\text{total}}(\theta)$, which updates the neural network parameters θ via gradient descent.

C Full MT-PINN Losses & Weighting (DWA)

C.1 Full Loss (by regime)

The total loss function for MT-PINN is expressed as a piecewise function depending on whether the risk-aversion is zero or positive:

$$\mathcal{L}_{\text{total}} = \begin{cases} w_{\text{PDE}}\mathcal{L}^{(0)}_{\text{PDE}} + w_{\text{traj}}\mathcal{L}^{(0)}_{\text{traj}} + w_{\text{IC}}\mathcal{L}^{(0)}_{\text{IC}} + w_{\text{sym}}\mathcal{L}^{(0)}_{\text{sym}}, & \lambda = 0, \\ w_{\text{PDE}}\mathcal{L}^{(\lambda)}_{\text{PDE}} + w_{\text{traj}}\mathcal{L}^{(\lambda)}_{\text{traj}} + w_{\text{IC}}\mathcal{L}^{(\lambda)}_{\text{IC}} + w_{\text{sym}}\mathcal{L}^{(\lambda)}_{\text{sym}} + w_{\text{0-term}}\mathcal{L}_{\text{0-term}}, & \lambda > 0. \end{cases}$$

PDE loss Let Γ_θ denote the value function from the network, and its partial derivatives are $\Gamma_\tau = \frac{\partial\Gamma_\theta}{\partial\tau}$, $\Gamma_X = \frac{\partial\Gamma_\theta}{\partial X}$, $\Gamma_S = \frac{\partial\Gamma_\theta}{\partial S}$, and $\Gamma_{SS} = \frac{\partial^2\Gamma_\theta}{\partial S^2}$. The PDE residual uses the HJB form according to

the regime:

$$\text{For } \lambda = 0: \ \mathcal{R}^{(0)}_{\text{HJB}} = \Gamma_\tau - \kappa^2 X^2 + \frac{1}{4}\Gamma_X^2, \quad \text{(Proposition A.1)}$$

$$\text{For } \lambda > 0: \ \mathcal{R}^{(\lambda)}_{\text{HJB}} = \Gamma_\tau - \frac{1}{2}\sigma^2 S^2 \Gamma_{SS} - \kappa^2 X^2 - \lambda S X + \frac{1}{4}\Gamma_X^2. \quad \text{(Proposition A.2)}$$

$\mathcal{L}^{(\cdot)}_{\text{PDE}}$ is then defined as:

$$\mathcal{L}^{(0)}_{\text{PDE}} = \frac{1}{N^{(0)}_{\text{PDE}}} \sum_{i=1}^{N^{(0)}_{\text{PDE}}} \left[\mathcal{R}^{(0)}_{\text{HJB}}(\tau_i, X_i)\right]^2, \quad (\tau_i, X_i) \sim \mathcal{D}^{(0)}_{\text{PDE}},$$

$$\mathcal{L}^{(\lambda)}_{\text{PDE}} = \frac{1}{N^{(\lambda)}_{\text{PDE}}} \sum_{i=1}^{N^{(\lambda)}_{\text{PDE}}} \left[\mathcal{R}^{(\lambda)}_{\text{HJB}}(\tau_i, X_i, S_i)\right]^2, \quad (\tau_i, X_i, S_i) \sim \mathcal{D}^{(\lambda)}_{\text{PDE}},$$

Inventory-zero loss (internal condition at $X = 0$. The loss of the internal condition at $X = 0$, defined by (2) for the risk-neutral regime and (7) for the risk-averse regime, is as follows:

$$\mathcal{L}^{(0)}_{IC} \approx \frac{1}{N^{(0)}_{IC}} \sum_{i=1}^{N^{(0)}_{IC}} \Gamma_\theta(\tau_i, 0)^2, \ \ \tau_i \sim \mathcal{D}^{(0)}_{IC}$$

$$\mathcal{L}^{(\lambda)}_{IC} \approx \frac{1}{N^{(\lambda)}_{IC}} \sum_{i=1}^{N^{(\lambda)}_{IC}} \left[\max\{\Gamma_\theta(\tau_i, 0, S_i), 0\}\right]^2, \ \ \tau_i, S_i \sim \mathcal{D}^{(\lambda)}_{IC}.$$

Symmetry loss. The symmetric condition loss encodes Corollary A.2.1 for the risk-neutral regime and Corollary A.2.2 for the risk-averse regime as:

$$\mathcal{L}^{(0)}_{\text{sym}} \approx \frac{1}{N^{(0)}_{\text{sym}}} \sum_{i=1}^{N^{(0)}_{\text{sym}}} \left[\Gamma_\theta(\tau_i, X_i) - \Gamma_\theta(\tau_i, -X_i)\right]^2, \ \ \tau_i, X_i \sim \mathcal{D}^{(0)}_{\text{PDE}},$$

$$\mathcal{L}^{(\lambda)}_{\text{sym}} \approx \frac{1}{N^{(\lambda)}_{\text{sym}}} \sum_{i=1}^{N^{(\lambda)}_{\text{sym}}} \left[\Gamma_\theta(\tau_i, X_i, S_i) - \Gamma_\theta(\tau_i, -X_i, -S_i)\right]^2, \ \ \tau_i, X_i, S_i \sim \mathcal{D}^{(\lambda)}_{\text{PDE}}.$$

Terminal zero-inventory loss. Due to the internal condition imposed on the risk-averse regime of (7), a separate loss term was created to ensure the network learns that $\Gamma(0, 0, S) = 0$ defined as:

$$\mathcal{L}_{\text{0-term}} \approx \frac{1}{N_{\text{term}}} \sum_{i=1}^{N_{\text{term}}} \Gamma(\tau = 0, X = 0, S_i)^2, \quad S_i \sim \mathcal{D}_{\text{term}}.$$

Note that the risk-neutral regime already explicitly imposes this constraint on all (τ, S) for $X = 0$.

Sampler definitions. $\mathcal{D}$ denotes the uniform sampling distribution for each collocation domain. Experiment-specific grids and counts (i.e., $N_{(\cdot)}$, horizon sets, and initial state grids) are provided in the respective experiment appendices.

C.2 DWA-style weight rebalancing

We adapt task weights with a dynamic weight average (DWA)-style scheme [7]. Instead of using DWA's raw ratio $L_{\mathcal{T}}(t-1)/L_{\mathcal{T}}(t-2)$ and softmax, we use a smoothed exponential moving average (EMA) of each loss and update weights from its relative scale. Concretely, we make the following lightweight stabilizations:

- EMA smoothing of losses, relative to initial scaling
- Geometric mean centering across task
- Gentle inverse power-law update instead of softmax and clip weights to $[w_{\min}, w_{\max}]$
- Mean-one normalization of weights across active tasks each update
- Optional freezing for tasks that become tiny over several updates
- Scheduled updates (i.e., updating weights every Δ epochs)

Hyperparameters are listed within the respective experiment appendices.

D Synthetic benchmark: complete experiment specification

The MT-PINN used for this experiment is outlined in Appendix C.1. As a replacement for the trajectory loss, which aims to naturally fulfill the terminal condition stated in equations (3) and (8), both PINN with λ-curriculum and vanilla PINN have a loss term that sets an arbitrarily large penalty for leftover inventory to fulfill the terminal condition. This loss term is formulated as:

$$\mathcal{L}_{\text{term}}^{(0)} = \frac{1}{N_{\text{term}}^{(0)}} \sum_{i=1}^{N} \left(\Gamma_\theta(0, X_i) - cX_i^2\right)^2, \;\; X_i \sim \mathcal{D}_{\text{term}}^{(0)},$$
$$\mathcal{L}_{\text{term}}^{(\lambda)} = \frac{1}{N_{\text{term}}^{(\lambda)}} \sum_{i=1}^{N} \left(\Gamma_\theta(0, X_i, S_i) - cX_i^2\right)^2, \;\; X_i, S_i \sim \mathcal{D}_{\text{term}}^{(\lambda)},$$

where $c > 0$ is the penalty strength. Otherwise the loss terms across the PINN variant models are the same:

$$\mathcal{L}_{\text{total}} = \begin{cases} w_{\text{PDE}}\mathcal{L}_{\text{PDE}}^{(0)} + w_{\text{IC}}\mathcal{L}_{\text{IC}}^{(0)} + w_{\text{sym}}\mathcal{L}_{\text{sym}}^{(0)} + w_{\text{term}}\mathcal{L}_{\text{term}}^{(0)}, & \lambda = 0, \\ w_{\text{PDE}}\mathcal{L}_{\text{PDE}}^{(\lambda)} + w_{\text{IC}}\mathcal{L}_{\text{IC}}^{(\lambda)} + w_{\text{sym}}\mathcal{L}_{\text{sym}}^{(\lambda)} + w_{\text{term}}\mathcal{L}_{\text{term}}^{(\lambda)}, & \lambda > 0. \end{cases}$$

For the synthetic benchmark experiment, three values of risk-aversion were evaluated: $\lambda^* \in \{0, 0.05, 0.1\}$. For $\lambda = 0$ all methods are trained for 30k epochs. For $\lambda > 0$, vanilla PINN was trained for 55k epochs, while the curriculum-based models used a two-phase schedule totaling 55k epochs (30k at $\lambda = 0$, then 5 curriculum stages of 5k epochs each up to the target λ^*).

Common setting (unless stated otherwise). Time horizon $T = 5.0$, state ranges $X \in [-10, 10], S \in [10, 100]$ (the S range only used when $\lambda > 0$). The PDE collocation points were $N_{PDE} = 30,000$ and internal conditions points were $N_{IC} = 5000$ (see 4). Model parameters: $\kappa = 0.1$ (risk-adjusted inventory penalty), $\sigma = 0.1$ (price volatility). Optimization uses AdamW with learning rate 5×10^{-4}. Loss weights are adapted with DWA-style scheme as briefly described in Appendix C.2 with weights updated every $\Delta = 1000$ epochs, weight clipping $[0.1, 2.0]$, EMA smoothing factor $\beta = 0.95$, update strength $\alpha = 0.3$, and freeze tolerance of 10^{-4}. Initial weights were set to $[w_{\text{PDE}} = 1.0, w_{\text{traj}} = 1.0, w_{\text{IC}} = 0.1, w_{\text{sym}} = 0.5, w_{\text{0-term}} = 0.5, w_{\text{term}} = 1.0]$.

Vanilla PINN (single-stage). MLP width of (500, 500, 500). For $\lambda = 0$: 30k epochs. For $\lambda \in \{0.05, 0.10\}$: 55k epochs with the same sampler sizes and network input dimensions were set by the regime. Terminal points were $N_{\text{term}} = 5000$.

PINN + λ-Curriculum (warm-start 1D state to 2D state). Phase A (1D, $\lambda = 0$): MLP width of (500, 500, 500), two input dimensions, and run for 30k epochs. Phase B (2D, $\lambda > 0$): warm start the trained 1D parameters to 2D then train 5 curriculum stages with $\alpha \in (0.25, 0.5, 0.75, 0.9, 1.0)$. Each stage runs 5k epochs, with $\lambda_\alpha = \alpha \cdot \lambda^*$. Terminal points were $N_{\text{term}} = 5000$.

MT-PINN + λ-Curriculum (warm-start 1D state to 2D state). Phase A (1D, $\lambda = 0$): MLP width of (32, 32, 32), two input dimensions, and run for 30k epochs. Phase B (2D, $\lambda > 0$): warm start the trained 1D parameters to 2D then train 5 curriculum stages with $\alpha \in (0.25, 0.5, 0.75, 0.9, 1.0)$. Each stage runs 5k epochs, with $\lambda_\alpha = \alpha \cdot \lambda^*$. Multi-trajectory batch samples $P = 820$ ($n_X = 41$ and where applicable $n_S = 20$) and roll out horizons of $\{T/50, T/10, T/5, 2T/5, 3T/5, 4T/5, T\}$ with $N_{\text{dt}} = 200$ Euler steps. Terminal points were $N_{\text{term}} = 2000$.

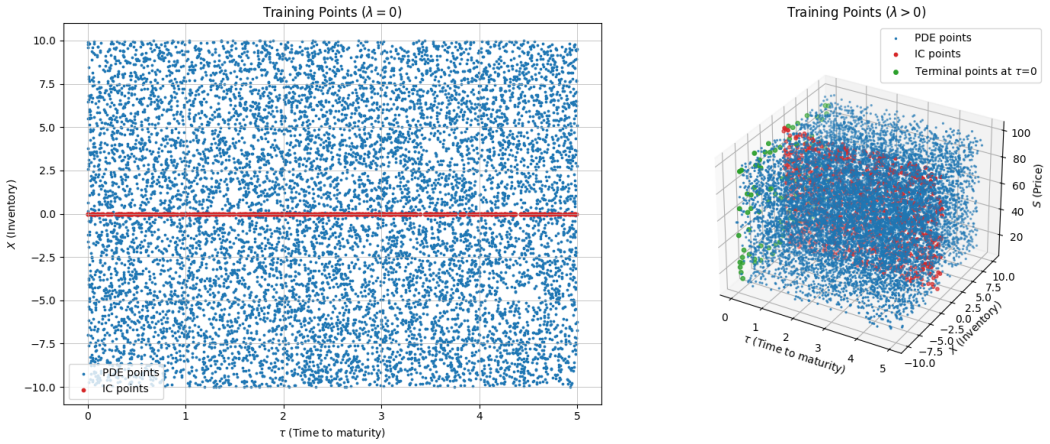


Figure 4: Sampling/collocation points

E Real-market SPY backtest: data, preprocessing, metrics, & experiment specification

E.1 Data & preprocessing

Intraday market data for SPY was obtained via Databento, sourced from the Nasdaq TotalView-ITCH feed. Data was used under Databento's academic/historical data provisions, subject to Nasdaq TotalView-ITCH license/ToS. We cannot redistribute the raw data.

The trading day is split into three 2-hour fixed intraday windows on 5-second intervals, and repeated across seven days (Feb 10 - Feb 19, 2025, excluding weekends). The windows are:

- Window 1 (W1): 09:45 - 11:45 ET
- Window 2 (W2): 11:45 - 13:45 ET
- Window 3 (W3): 13:45 - 15:45 ET

This intraday segmentation of SPY mid-price is illustrated in Figure 5. Note that these windows exclude the first 15 minutes of market open and the last 15 minutes of market close. This was done to avoid the noisy and elevated volatilities typically observed at those times [6].

E.2 Metrics

The main criteria for comparing these models were their exposure and cost trade-off. Exposure measures how much inventory the strategy was carrying on average for a given window. This was calculated using

$$\text{Exposure}(W_j) = \frac{1}{N}\sum_{k=0}^{N-1}\left(\chi_k^{(j)}\right)^2,$$

where W_j is the j^{th} trading window, N is the number of discretized equal steps, χ_k is the normalized inventory at step k. Cost is measured as the implementation shortfall relative to liquidating the entire initial position X_0 at the initial mid-price S_0. In practice, this measures the cost associated with using the execution model for liquidation versus dumping the entire position immediately at the current mid-price. Cost was measured in basis points (1 bps = 0.01%) and computed using

$$\text{Cost}(W_j) = \frac{S_0^{(j)} X_0 - \sum_{k=0}^{N-1} q_k^{(j)} S_k^{(j)}}{S_0^{(j)} X_0} \times 10^4 \ \text{ bps},$$

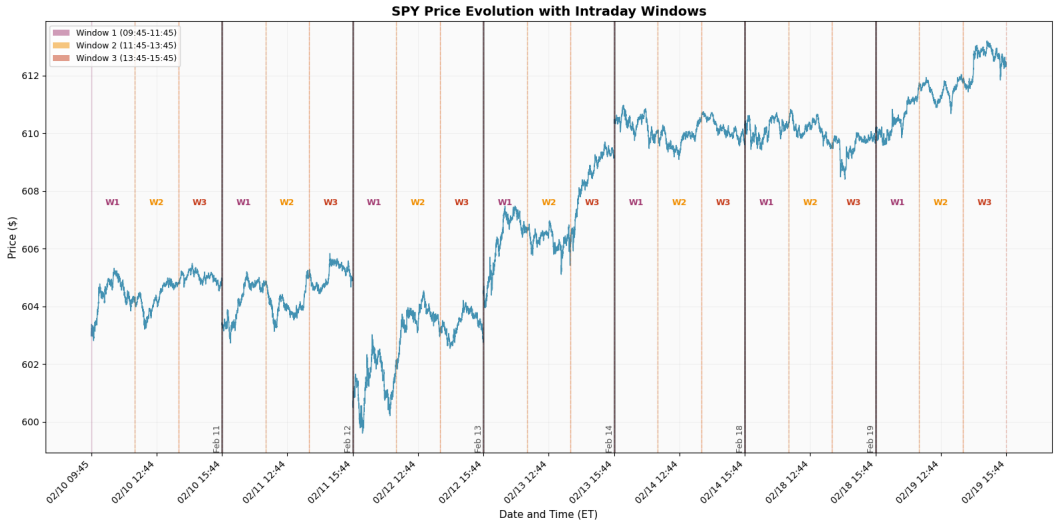


Figure 5: Intraday segmentation of SPY mid-price series into fixed windows. Each trading day is partitioned into three non-overlapping 2-hour intervals: Window 1 (W1) from 09:45 to 11:45 ET, Window 2 (W2) from 11:45 to 13:45 ET, and Window 3 (W3) from 13:45 to 15:45 ET. Vertical dashed lines mark the boundaries between windows, while solid black vertical lines indicate the start of each new trading day.

where q_k denotes the trade at step k. The exposure and cost across all the trading windows are then aggregated to report the mean and standard deviation of the exposure and cost using NumPy built-in functions (see Table 2 for results).

E.3 Experiment specification

Part of applying these models into real-market data meant adjusting the model parameters so that they accurately match the data.

Time Normalization. Time horizon was expressed by trading day units, where elapsed time is normalized by the effective length of a trading day on the U.S. equity markets (6.5 hours, from 9:30 a.m. to 4:00 p.m. EST). Specifically, the two-hour backtest horizon corresponded to

$$T_{\text{days}} = \frac{2}{6.5} \approx 0.308,$$

which represents roughly 31% of the trading day.

Inventory Normalization. Inventory was normalized by the initial position, so that the inventory lies in the interval $[-1, 1]$, and represents the fraction of the initial inventory:

$$\chi_t = \frac{x_t}{X_0},$$

where $\chi_t = 1$ corresponds to holding the full initial inventory, $\chi_t = 0$ to fully liquidated inventory, and $\chi_t = -1$ corresponds to over-liquidation (shorting) of equal size of the initial inventory.

Stock Price Range. Over the full 21 time windows, the stock price range was: $\$599.60 - \$613.20$. Therefore, the S range used for MT-PINN was $590 - 620$, leaving a small buffer of roughly 1% each side.

Volatility Estimation. To estimate volatility (σ) of the assumed unaffected price process S_t, the realized volatility for each intraday execution window was computed and then averaged. Given a set of stock mid-prices $\{S_{t_i}\}_{i=1}^{N}$ in a given window of length ΔT_{W}, the log-returns can be calculated as

$$r_i = \ln\left(\frac{S_{t_i}}{S_{t_{i-1}}}\right), \quad i = 1, ..., N.$$

Hence the realized volatility for a given window is

$$\hat{\sigma}^2_{\mathrm{W}} = \frac{1}{N-1}\sum_{i=1}^{N}(r_i - \bar{r})^2,$$

where $\bar{r}$ denotes the sample mean of returns. For window length ΔT_W, realized volatility was rescaled into trading-day units as

$$\hat{\sigma}_{\mathrm{W,\,daily}} = \frac{\hat{\sigma}_{\mathrm{W}}}{\sqrt{\Delta T_{\mathrm{W}}}}.$$

Finally, averaging across all execution windows yields the representative daily volatility parameter for the simulation:

$$\sigma = \frac{1}{J}\sum_{j=1}^{J}\hat{\sigma}^{(j)}_{\mathrm{W,\,daily}},$$

where J represents the number of execution windows. From the SPY market data, the calculated volatility was:

$$\sigma \approx 0.0038 \quad (\approx 6\% \text{ annualized}).$$

MT-PINN (Phase A: 1D, Phase B: 2D). MT-PINN with width (32,32,32), scalar output, and tanh activations. Time horizon $T \approx 0.308$, with horizon grid $T_j \in \{T/20, T/10, T/4, T/2, 3T/4, T\}$, $Ndt_{\text{traj}} = 50$ Euler steps and $n_X = 200$ inventory grid points. Inventory range set to $[-1, 1]$ and $\kappa = 0.2$. This experiment uses the same hyperparameters for the weights as outlined in Appendix D, with the exception to weighting clipping $[0.1, 5.0]$ and initial trajectory loss weighting $w_{\text{traj}} = 5.0$.

Phase A: uses (τ, X) inputs, 20k training epochs with 30k collocation points, and risk-neutral regime $\lambda = 0$. Phase B: uses (τ, X, S) with price $S \in [590, 620]$, discretized with $n_S = 41$ for multi-trajectory loss, and trained on 20k collocation points for 5k epochs. Volatility set to $\sigma = 0.0038$. Risk aversion parameter introduced through curriculum stages $\alpha \in \{0.25, 0.5, 0.75, 1.0\}$ for $\lambda^* = [0.05, 0.1]$, with $\lambda_\alpha = \alpha \cdot \lambda^*$. Sample sizes: $N_{IC} = 2000, N_{\text{0-term}} = 200$.

F Further Synthetic Benchmark & SPY Real-Market Backtest Results

F.1 Synthetic Benchmark Results

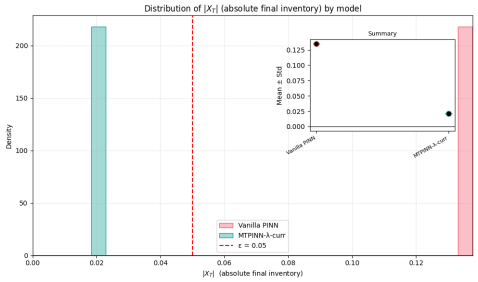
For $\lambda = 0$ only vanilla PINN and MT-PINN-λ-curriculum are shown as PINN-λ-curriculum collapses to vanilla PINN and is omitted.

Figure 6 shows the histogram of the absolute terminal inventory $|X_T|$ across 200 stock price simulations with an ϵ tolerance market and mean±std inset of the terminal inventory. It is evident that the MT-PINN-λ-curriculum concentrates heavily near zero and attains the highest pass-rate $p_\epsilon = Pr(|X_T| \leq \epsilon)$ for every λ (see Table 3). Baselines show larger violations for zero terminal inventory, with the PINN-λ-curriculum exhibiting improvements over the vanilla PINN for larger λ.

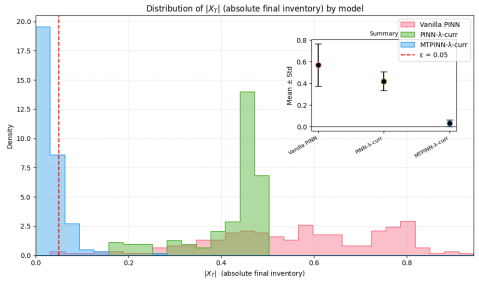
Figure 8 illustrates the implied trading rate $v^*(t) = \frac{1}{2}\partial_X \Gamma_\theta$ versus the analytical solution derived by Gatheral and Schied for all three paths. MT-PINN-λ-curriculum follows closer to the closed-form across the horizon for all λ, while the baselines show higher variance.

Table 3: Absolute terminal-inventory enforcement on the synthetic benchmark for 200 simulated price paths for varying λ. It is reported $p_\varepsilon = \Pr(|X_T| \leq \varepsilon)$ with $\varepsilon = 0.05$.

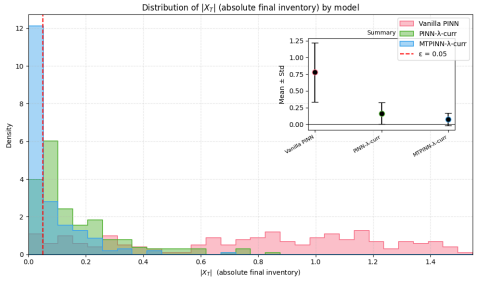
	$\lambda = 0$			$\lambda = 0.05$			$\lambda = 0.10$		
Method	Mean±Std	95th pct	p_ε	Mean±Std	95th pct	p_ε	Mean±Std	95th pct	p_ε
Vanilla PINN	0.135±0.000	0.135	0.000	0.570±0.196	0.826	0.000	0.777±0.444	1.407	0.055
PINN-λ-curr	-	-	-	0.420±0.085	0.481	0.000	0.164±0.161	0.527	0.205
MT-PINN-λ-curr	**0.022±0.000**	**0.022**	**1.000**	**0.031±0.030**	**0.072**	**0.810**	**0.073±0.092**	**0.241**	**0.600**



(a) $\lambda = 0$

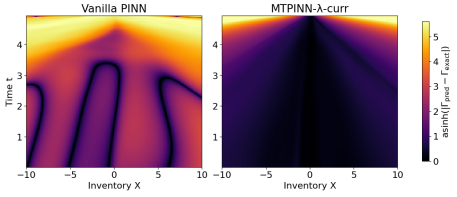


(b) $\lambda = 0.05$

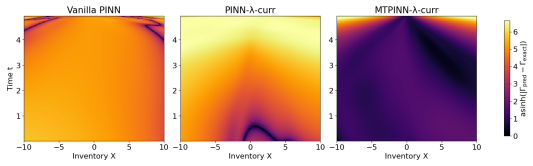


(c) $\lambda = 0.10$

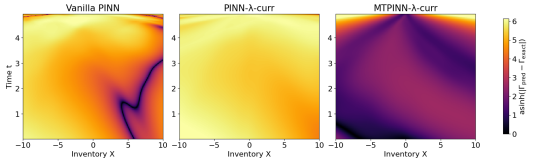
Figure 6: Distribution of $|X_T|$ (absolute final inventory) by model for each risk aversion λ. Each panel shows the histogram across rollouts for the three models (Vanilla PINN, PINN-λ-curr, MTPINN-λ-curr; PINN-λ-curr omitted for $\lambda = 0$) with the tolerance line at $\epsilon = 0.05$ and the inset reporting mean $\pm$std.



(a) $\lambda = 0$

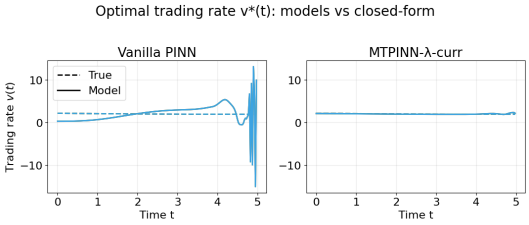


(b) $\lambda = 0.05$

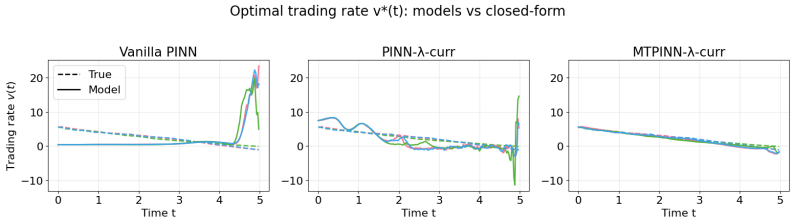


(c) $\lambda = 0.10$

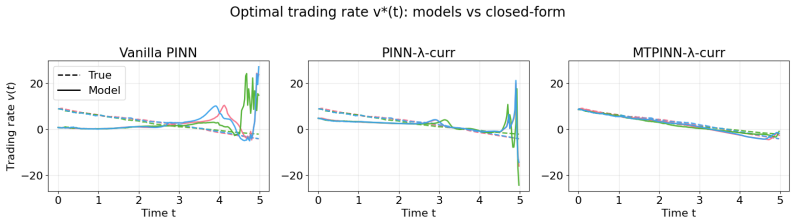
Figure 7: Heatmaps of domain-wide error over (t, X) for each risk aversion λ with fixed $S = 55$. Each panel compares methods as in the per-λ heatmap export.



(a) $\lambda = 0$



(b) $\lambda = 0.05$

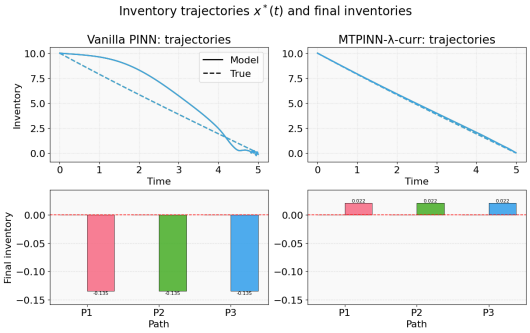


(c) $\lambda = 0.10$

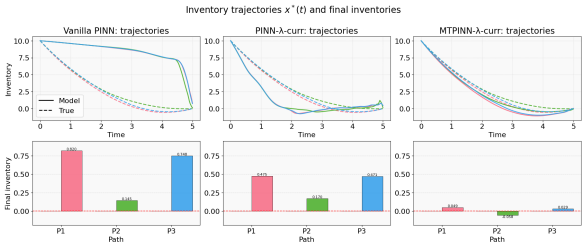
Figure 8: Optimal trading rate $v^*(t)$: model vs. closed form for each λ.

Table 4: Error statistics for Vanilla PINN, PINN-λ-curr, and MT-PINN-λ-curr across $\lambda \in \{0, 0.05, 0.10\}$ (with PINN-λ-curr omitted at $\lambda = 0$) and fixed stock price $S = 55$. Mean absolute error, max absolute error, mean relative error, and root-mean squared error are reported in both original space and arcsinh-transformed space.

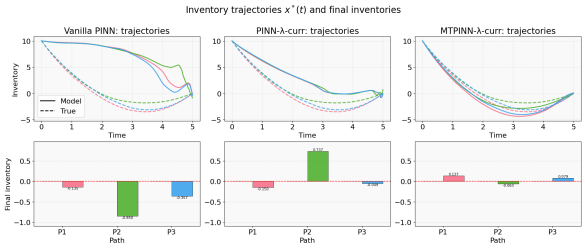
λ	Method	Original Space				Arcsinh Space			
		MAE	MaxAE	MRE	RMSE	MAE	MaxAE	MRE	RMSE
0	Vanilla PINN	22.75	**363.18**	7.43	**45.45**	0.80	4.96	3.66	1.07
	PINN-λ-curr	-	-	-	-	-	-	-	-
	MT-PINN-λ-curr	**16.44**	1773.53	**0.24**	116.00	**0.12**	**2.18**	**0.18**	**0.25**
0.05	Vanilla	78.98	573.98	33.69	**88.12**	3.97	10.03	2.80	5.29
	PINN-λ-curr	145.77	**413.17**	62.52	192.12	3.49	9.30	2.66	4.51
	MT-PINN-λ-curr	**17.73**	1756.74	**0.77**	115.68	**0.28**	**2.10**	**0.51**	**0.50**
0.10	Vanilla PINN	81.22	475.45	33.92	**99.78**	4.33	9.71	2.27	5.61
	PINN-λ-curr	140.12	**270.59**	44.04	147.14	5.23	11.02	2.60	6.57
	MT-PINN-λ-curr	**19.65**	1742.84	**1.09**	115.39	**0.37**	**3.84**	**0.56**	**0.73**



(a) $\lambda = 0$

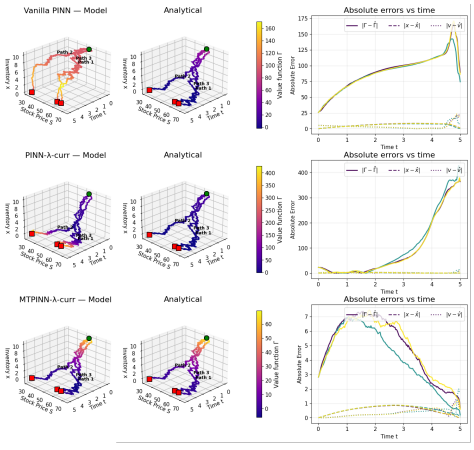


(b) $\lambda = 0.05$

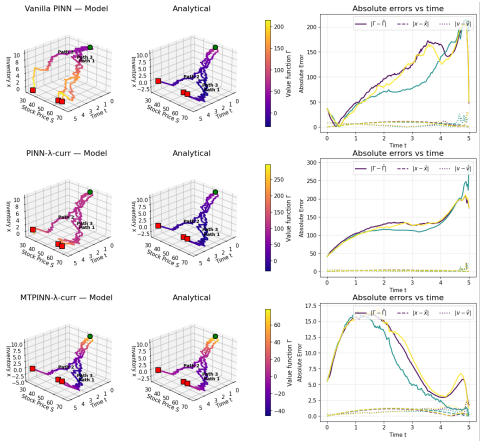


(c) $\lambda = 0.10$

Figure 9: Inventory trajectories $x^*(t)$ and final inventories across paths for each λ.

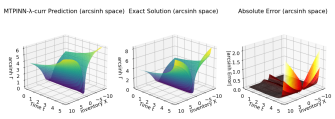


(a) $\lambda = 0.05$

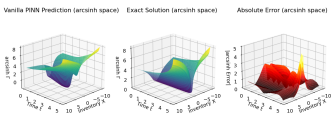


(b) $\lambda = 0.10$

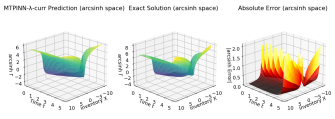
Figure 10: Pathwise rollouts: model vs. analytic trajectories and absolute error over time for each $\lambda > 0$.



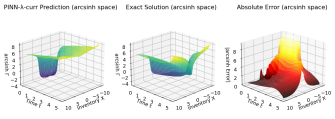
(a) MT-PINN-λ-curr, $\lambda = 0$



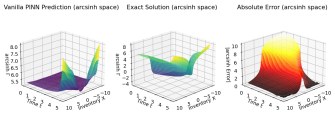
(b) Vanilla PINN, $\lambda = 0$



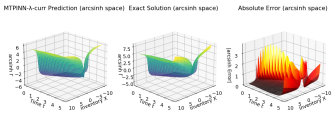
(c) MT-PINN-λ-curr, $\lambda = 0.05$



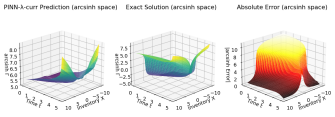
(d) PINN-λ-curr, $\lambda = 0.05$



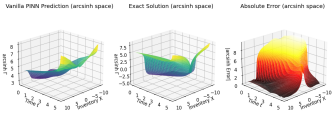
(e) Vanilla PINN, $\lambda = 0.05$



(f) MT-PINN-λ-curr, $\lambda = 0.10$



(g) PINN-λ-curr, $\lambda = 0.10$



(h) Vanilla PINN, $\lambda = 0.10$

Figure 11: Arcsinh-space value surfaces (each panel is organized as: Prediction / Exact / Absolute Error). Rows vary λ, columns vary the method.

Figure 9 reveals the inventory trajectories $x^*(t)$ under the learned trading rate for all three paths. Although MT-PINN-λ-curriculum's accuracy in matching the closed-form solution reduces as λ increases, it consistently produces near zero-inventory at terminal and nevertheless outperforms the baseline PINNs. The PINN-λ-curriculum improves on the vanilla PINN for higher λ, somewhat grasping the concept of reduced risk exposure, unlike the vanilla PINN which increases its risk exposure.

Figure 10 displays the models versus analytical inventory trajectories for three paths, showing the value function along the path and the absolute error over time for the value function, inventory trajectory, and trading rate. Note that the plots only show for $\lambda > 0$, because, as previously stated, for $\lambda = 0$, the optimal control and trajectories are invariant to the stock price S. MT-PINN-λ-curriculum shows a reduction in nearer terminal errors, reduced path-variance, and orders of magnitude smaller error than the baselines.

Figure 7 exhibits the domain-wide inverse hyperbolic sine function absolute value function error $(\text{asinh}(|\Gamma_{\text{pred}} - \Gamma_{\text{exact}}|))$ over (t, X) for fixed mid-range stock price $S = 55$. MT-PINN-λ-curriculum has reduced error across the domain with higher error near terminal, as can be seen in Table 4. Table 4 shows that across the arcsinh space, MT-PINN-λ-curriculum performs the best across all metrics. In the original space, MT-PINN-λ-curriculum performs best in mean absolute error and mean-relative error, while PINN-λ-curriculum consistently performed best in root-mean-squared error and, for the most part, maximum absolute error. The rationale for why MT-PINN-λ-curriculum performs better in the arcsinh space than it does in the original space is because the errors in the outliers, mostly found near maturity, aren't penalized as much in the arcsinh space. This demonstrates that the baseline PINNs capture the singularity in the value function with less error, however, overall, MT-PINN-λ-curriculum better captures the value function across the entire domain.

Figure 11 provides a 3D view of Figure 7. It is evident that, domain-wide, the baseline PINNs have less error near terminal than the MT-PINN-λ-curriculum due to their terminal condition loss; however, as has been outlined, this doesn't result in accurate control policy.

F.2 SPY Real-Market Backtest Results

Figure 12 and Table 2 both show the cost-variance/exposure trade-offs. It is clear that the MT-PINN is behaving well for varying λ in the sense that for higher λ, the model is experiencing less exposure and for $\lambda = 0$ the model behaves very similarly to TWAP, both as expected. For this given SPY market data, TWAP/risk-neutral MT-PINN experience lower costs, likely as a result of rising SPY market prices across the sampled time windows. This is evidenced in Figure 13, where the execution trajectories and costs for the MT-PINN with different λ and TWAP across four chosen trading

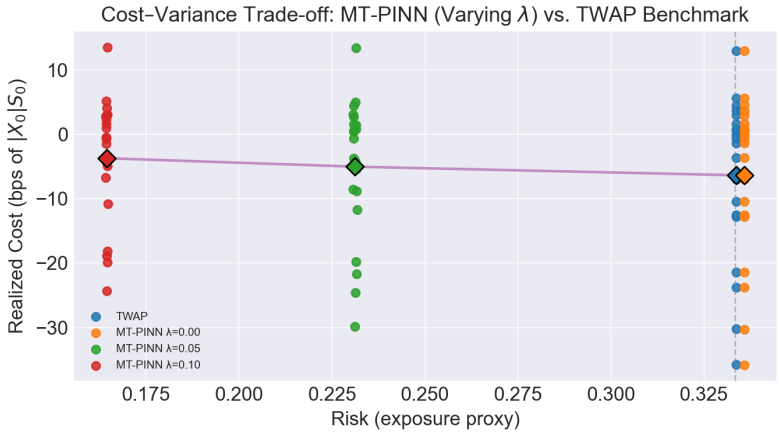


Figure 12: Cost-variance/exposure trade-off of MT-PINN for varying $\lambda \in [0, 0.05, 0.1]$ and TWAP.

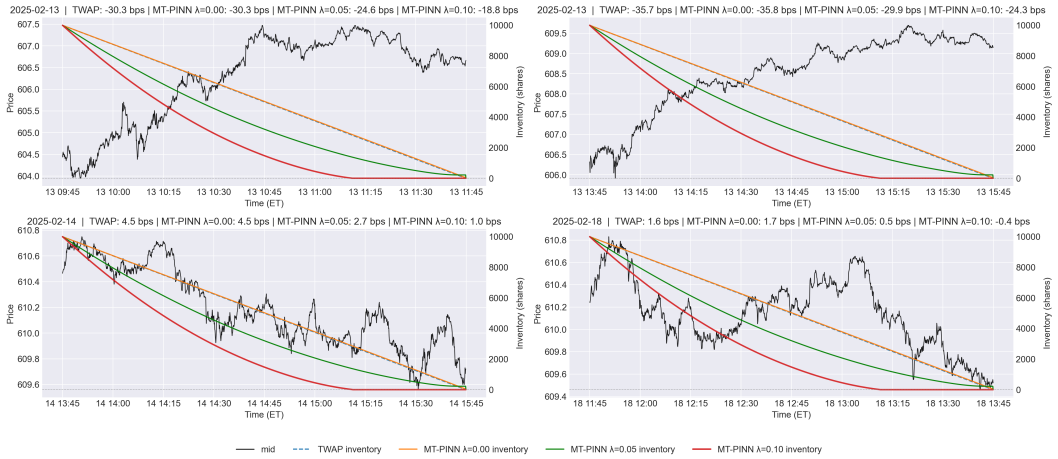
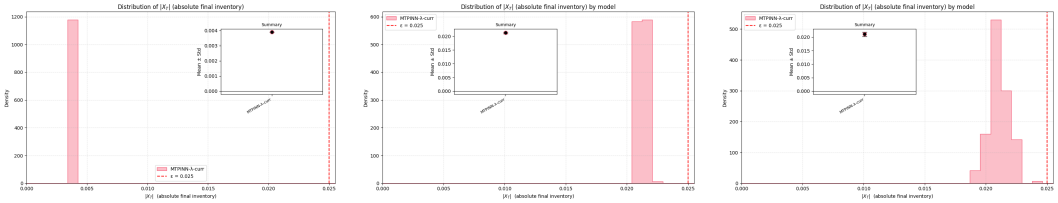


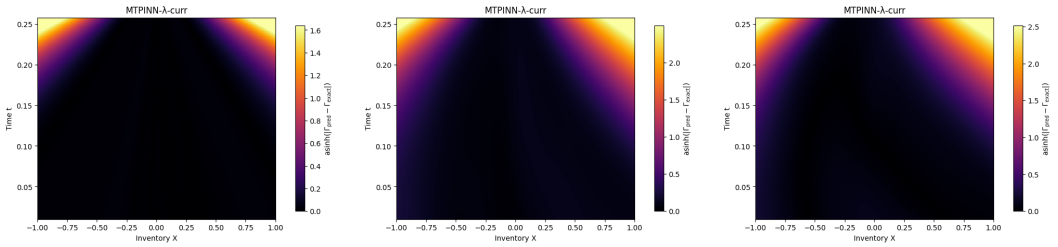
Figure 13: Execution trajectories and costs for MT-PINN with varying $\lambda \in [0, 0.05, 0.1]$ versus TWAP over four sampled trading windows of the SPY ($[W_{10}, W_{12}, W_{15}, W_{17}]$). The top two subplots show rising SPY windows, while the bottom two subplots show falling SPY windows.

windows. The top two time window plots show rising SPY prices and report lowest costs associated with the TWAP algorithm. In contrast, the bottom two time window plots show falling SPY prices and detail lowest costs associated with the highest risk-averse MT-PINN. It is important to mention that this backtest assumed no shorting/over-liquidation and ensures that once zero-inventory is achieved, no subsequent trades are made.



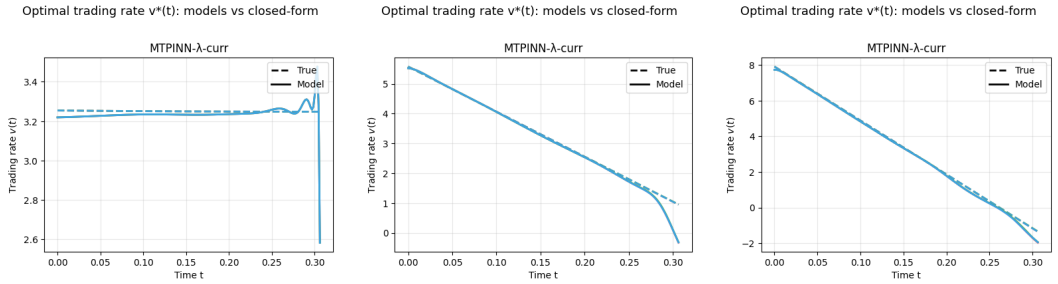
(a) $\lambda = 0$ (b) $\lambda = 0.05$ (c) $\lambda = 0.10$

Figure 14: Distribution of $|X_T|$ (absolute final inventory) for each risk aversion λ. Each panel shows the histogram across rollouts for the MT-PINN model with the tolerance line at $\epsilon = 0.025$ and the inset reporting mean $\pm$std.



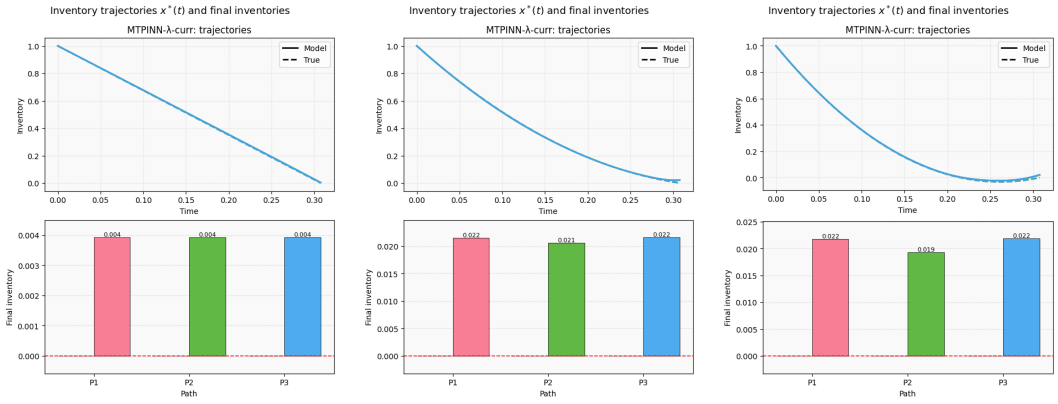
(a) $\lambda = 0$ (b) $\lambda = 0.05$ (c) $\lambda = 0.10$

Figure 15: Heatmaps of domain-wide error over (t, X) for each risk aversion λ with fixed $S = 605$.



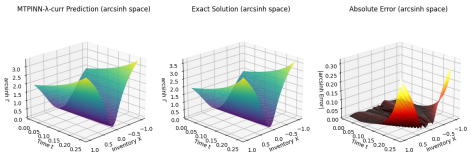
(a) $\lambda = 0$ (b) $\lambda = 0.05$ (c) $\lambda = 0.10$

Figure 16: Optimal trading rate $v^*(t)$: model vs. closed form for each λ.

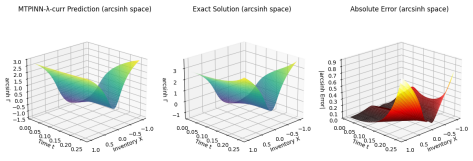


(a) $\lambda = 0$ (b) $\lambda = 0.05$ (c) $\lambda = 0.10$

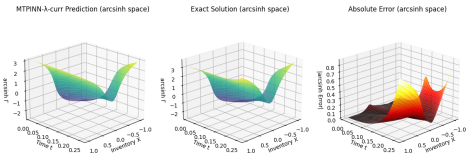
Figure 17: Inventory trajectories $x^*(t)$ and final inventories across paths for each λ.



(a) $\lambda = 0$

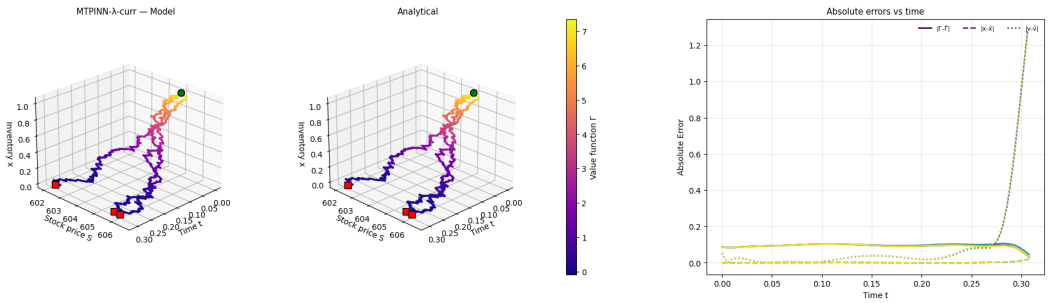


(b) $\lambda = 0.05$

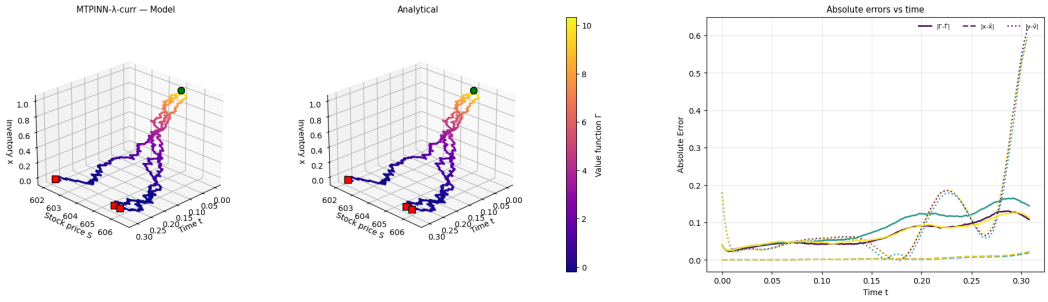


(c) $\lambda = 0.10$

Figure 18: Arcsinh-space value surfaces for each λ.



(a) $\lambda = 0.05$



(b) $\lambda = 0.10$

Figure 19: Pathwise rollouts: model vs. analytic trajectories and absolute error over time for each $\lambda > 0$.

Figure 14 further exemplifies MT-PINN's consistency with reaching near-zero terminal inventory, especially with lower λ values. Figure 15 exhibits the domain-wide inverse hyperbolic sine function absolute value function error ($\text{arcsinh}(|\Gamma_{\text{pred}} - \Gamma_{\text{exact}}|)$) over (t, X) for fixed stock price $S = 605$, showing low error across most of the domain with the exception to high terminal inventory. This is similarly illustrated in the arcsinh space value surfaces in Figure 18. Nonetheless, Figure 16 and 17 illustrates the MT-PINNs ability in closely matching the trading rate and inventory trajectory closed form solution across varying λ. Finally, Figure 19 displays the risk-averse MT-PINNs vs analytical inventory trajectories for three paths, showing the value function along the path and the absolute error over time for the value function, inventory trajectory, and trading rate.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract/introduction claim that (i) a trajectory loss results in less zero terminal inventory violations, (ii) a λ-curriculum stabilizes training, and (iii) synthetic and real-market data experiments validate the approach, which are all evidenced by Table 1 and 3, Figure 1 and Figure 2, and Table 2.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: §5 discusses fees omission, single-asset scope, simplified impact, and MT-PINN complexity.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Proposition A.1/A.2 with assumptions and proofs are in Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: §3-§4 largely cover most hyperparameters, sampling grids, curriculum schedule, metrics, while Appendix D and Appendix E the rest of it.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is available through `https://github.com/anthimevalin/Multi-Trajectory-PINNs-Zero-Terminal-HJB`. The SPY Data is from Databento (Nasdaq TotalView-ITCH) as noted in Appendix E.1.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and testing details (grids, horizons, λ schedules, optimizer, etc.) are in §3-4 and Appendices D and E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Mean $\pm$ standard derivation/error bars and percentiles are provided with explicit reference to the use of NumPy for such calculations in Appendix E.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: 1x v6e-1 TPU, JAX/FLAX, and typical runtimes are reported in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No human subjects, Databento market data usage described, and anonymity preserved.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: No direct deployment discussion.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No release of high-risk models/datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Databento discussed in Appendix E.1.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No released code, however, if requested, can be provided and with documentation included.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No human subjects or crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs only used for writing/editing/formatting.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.