Optimal Execution – From the Almgren–Chriss Model to Deep Reinforcement Learning

¡Your Name¿

¡Your Institution¿

July 15, 2025

Agenda

1. Motivation
2. Problem Formulation as an MDP
3. Reward Engineering
4. Enhanced Market Simulator
5. Algorithmic Benchmarks
6. Conclusion

Why Optimal Execution Matters

- Large orders move prices *(market impact)* – poor scheduling increases costs.
- Execution quality is traditionally measured by Implementation Shortfall (IS) and Expected Shortfall (ES).
- Classic solution: Almgren–Chriss (AC) closed-form optimal schedule under specific assumptions.
- But **market micro-structure is richer**: non-linear impact, stochastic liquidity, fees, etc.

Why Move Beyond Almgren–Chriss?

Limitations of AC

- Assumes arithmetic Brownian motion (ABM) or deterministic drift.
- Linear temporary and permanent impact.
- Risk captured only via variance of proceeds.
- Static schedule $\Rightarrow$ cannot react to intra-day price information.

Opportunity for RL

- Model-free: learn directly from simulated or historical LOB.
- Naturally handles high-dimensional states/actions.
- Can optimise non-linear objectives (e.g. *CVaR*).

State Design

Chosen state $s_k = \big(\underbrace{r_{k-D+1:k}}_{\textbf{log-returns}}, \underbrace{m_k}_{\textbf{time left}}, \underbrace{i_k}_{\textbf{inventory left}} \big)$

- $r_{k-D+1:k}$: window of $D = 5$ past log-returns (6 returns $\Rightarrow$ 6 periods) captures recent momentum and volatility.
- $m_k = \frac{T-k}{T}$: fraction of horizon remaining provides a natural clock.
- $i_k = \frac{Q_k}{Q_0}$: remaining inventory fraction informs risk of holding.

Is $D = 5$ **optimal?**

- Larger D offers richer autocorrelation signals but increases dimension & sample complexity.
- Smaller D reduces noise resilience.
- *Empirically*, $D \in [3, 8]$ showed marginal gains; tune via validation.

Action Space

- Continuous `a_k` $\in [0, 1]$: proportion of *current* inventory to sell at step k.
- Transformed to actual shares via $Q_{\text{sell}} = a_k \, Q_k$.
- Alternative: **actions as trading rates** $u_k \in \mathbb{R}_+$.

TODO: Add equations linking agent action to price impact (see Eq. (15) in `syntheticChrissAlmgren.py`).

Custom Reward Function

Objective

Minimise **Expected Shortfall ($\mathbf{ES}_\alpha$)** at risk level α while penalising variance.

$$R = -\mathrm{ES}_{0.95}(\mathrm{P\&L}) - \lambda_1\sigma^2 - \lambda_2\,\epsilon\,|Q_{\mathrm{sell}}|$$

- CVaR surrogate implemented via auxiliary variable η (see `ddpg`$_a$*gent.py*).
- `Trading fee` ϵ `appended to imitate real markets.`
- `Discount factor` $\beta = 0.9999$ `to emphasise early proceeds.`

Result: **TODO: Insert comparison table: AC vs DDPG + custom reward.**

Dense vs Sparse Rewards

- **Dense**: agent rewarded at each step based on instantaneous proceeds.
- **Sparse**: zero reward until terminal liquidation.

Dense vs Sparse Rewards

- **Dense**: agent rewarded at each step based on instantaneous proceeds.
- **Sparse**: zero reward until terminal liquidation.

Observation: Dense rewards accelerate convergence; sparse rewards encourage risk-aware behaviour but require longer training. **TODO: Add learning-curve figure here.**

Price Dynamics – Switching to GBM

- Replaced arithmetic Brownian motion with Geometric Brownian Motion (GBM):
$$dS_t = \mu S_t \mathrm{d}t + \sigma S_t \mathrm{d}W_t.$$
- Adjusted single-step variance in environment (see `syntheticChrissAlmgren.py`).

Impact on Policy

- Higher proportional variance $\Rightarrow$ larger tail-risk; CVaR term becomes more binding.
- Learned schedule skews toward *front-loading* when drift $\mu < 0$.

TODO: Insert PL distribution plot (GBM vs ABM).

Adding Trading Fees & Non-linear Impact

- Fixed fee ϵ already in AC; we add proportional fee $\lambda_2 \times v^2$.
- Encourages smoother execution path.

TODO: Show before/after utility comparison.

DDPG Baseline (Our Implementation)

- Actor–Critic with OU-noise exploration.
- Replay buffer size 10^4, batch size 128.
- Achieved $\mathrm{ES}_{0.95}$: **TODO: XX% improvement over AC.**

Space for Further Work

SAC and TD3 Experiments

- **TODO: Insert architecture + hyper-parameters**
- **TODO: Insert performance metrics vs DDPG**

Please complete once models are trained and evaluated.

Key Takeaways

- AC provides analytical insight but is rigid.
- Deep RL absorbs richer signals and objectives, outperforming AC on ES.
- Reward shaping and environment realism (GBM, fees) materially influence learned policy.
- Future: ensemble of SAC/TD3, calibration on LOB simulator (e.g. ABIDES).

References I

R. Almgren and N. Chriss. *Optimal Execution of Portfolio Transactions*. 2001.

J. Gatheral and A. Schied. *Optimal Trade Execution under GBM*. 2012.

P. Cheridito and M. Weiss. *Reinforcement Learning for Trade Execution with Market Impact*. arXiv:2507.06345, 2025.

Y. Hafsi and E. Vittori. *Optimal Execution with Reinforcement Learning*. arXiv:2411.06389, 2024.