Student: Bogdan Alexandrov
Research Advisor: Pavel Osinenko

# Limit Order Book Queue Modelling: a Reinforcement Learning Approach

MSc Program

Data Science

June, 2024

Skoltech

# Motivation

- At the moment, there are more than 50 funds worldwide engaged in high-frequency trading.
- One of the most important tasks for quantitative researchers is evaluating trading strategies.
- The profitability of the hft-funds depends on the accuracy of the evaluation of strategies.

# Background: orders

Market dynamics is a sequence of orders from participants.

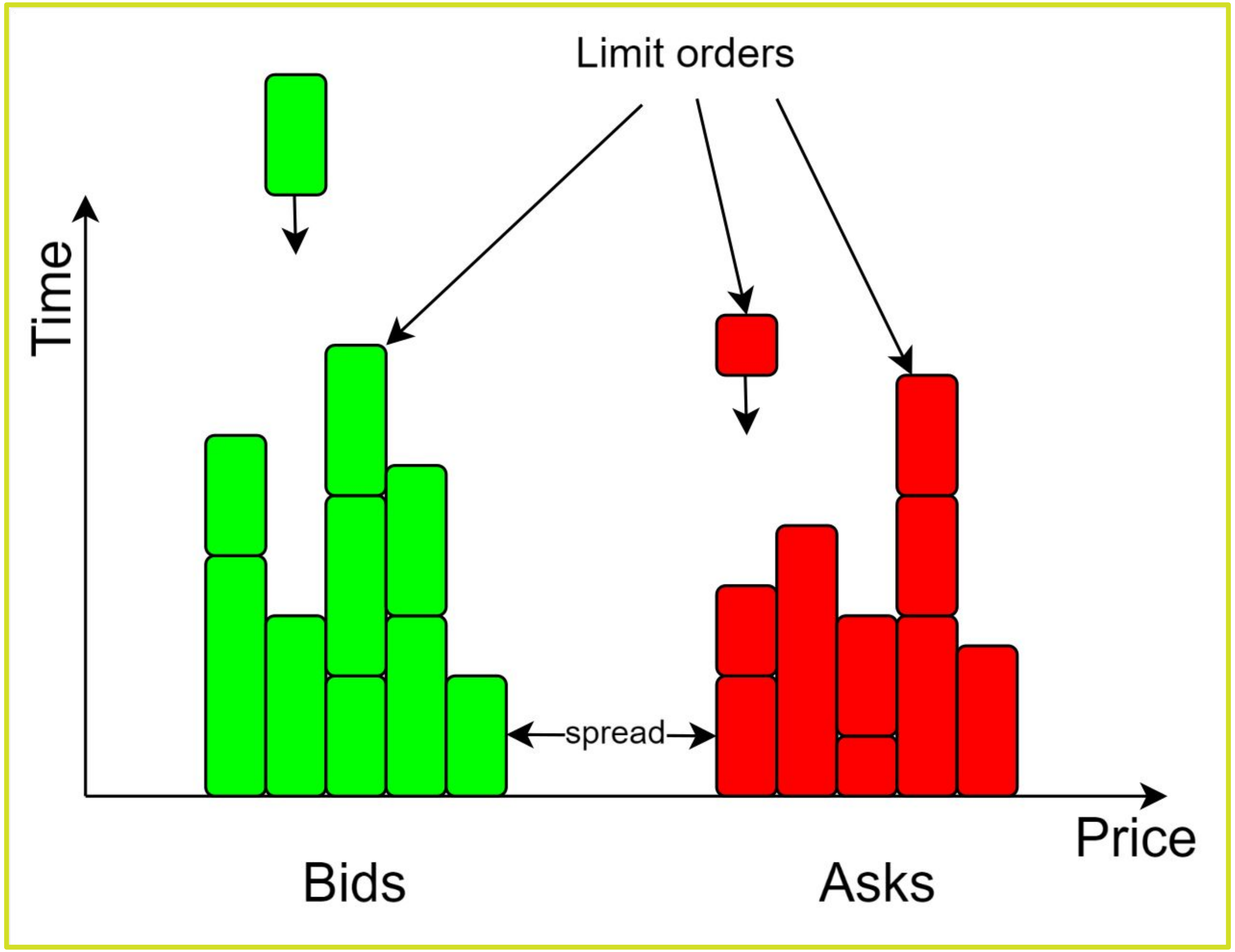
**Types of orders:**

- Limit order
- Market order
- Cancel order
- Modification order

Limit and market orders are also divided into 2 classes:

- Bid (buy)
- Ask (sell)

# Background: order book structure




Skoltech

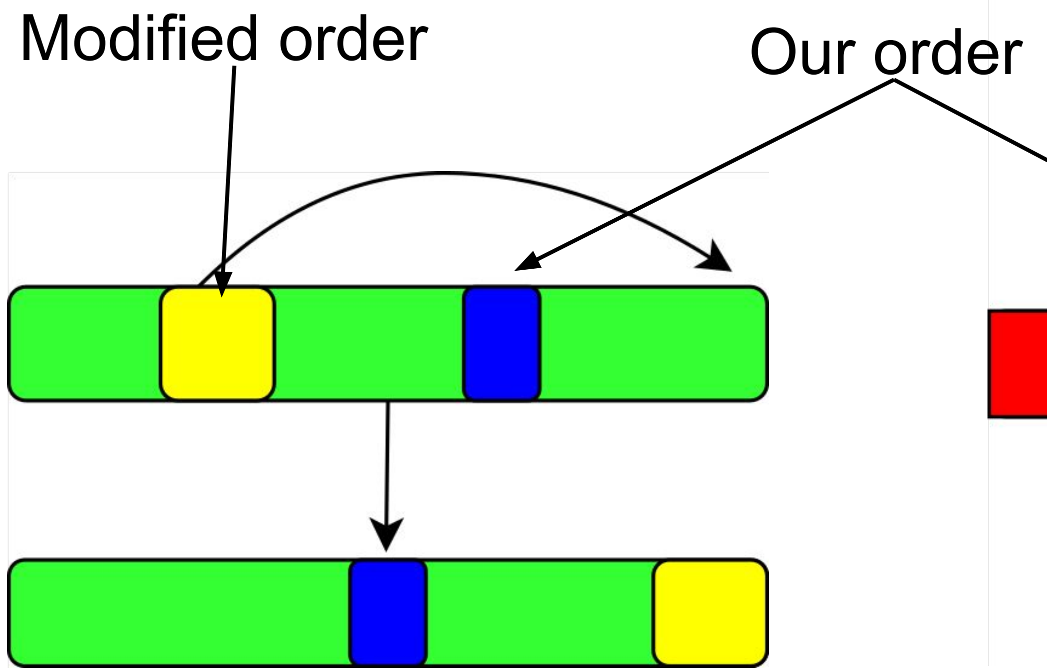
# Background: uncertainty in the backtest

**Why can't we evaluate the strategy with 100% accuracy?**

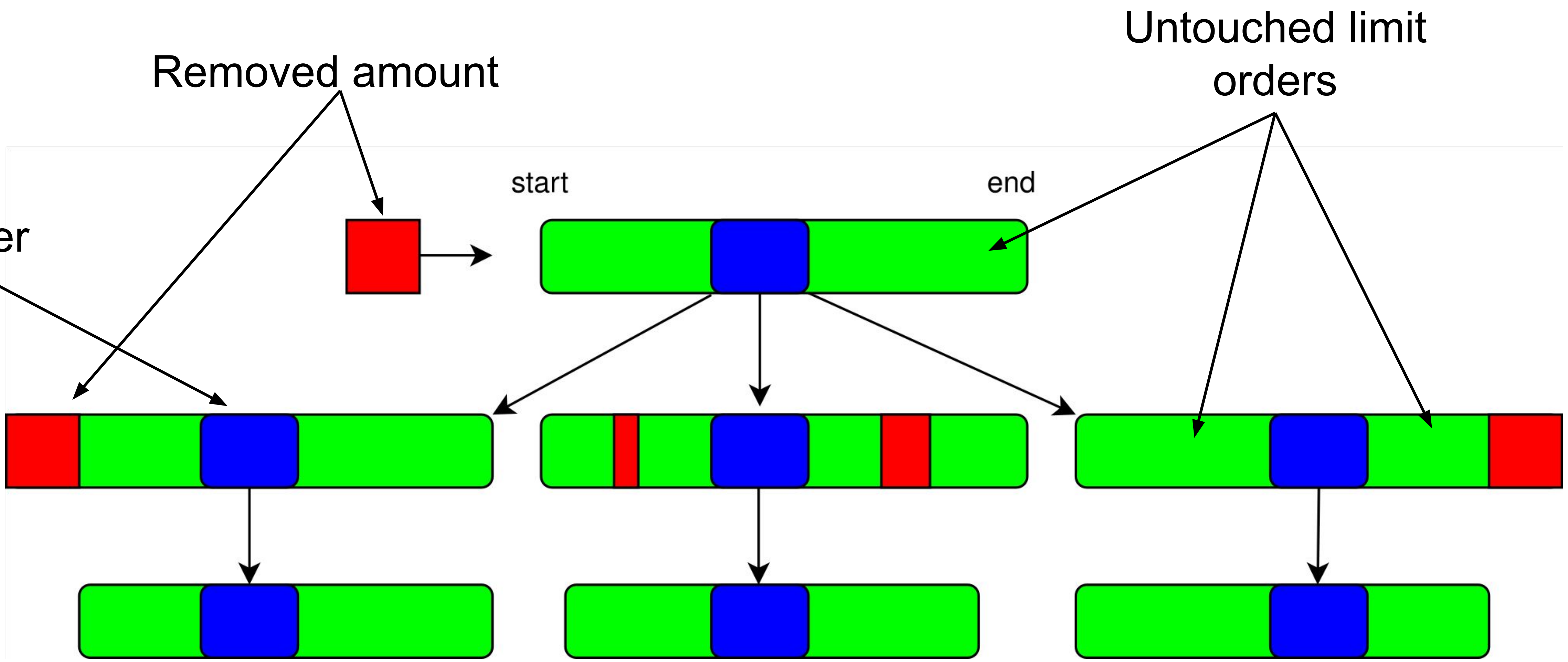
- **Latency**
  - Depends on the distance to the servers
- **Market Impact**
  - For small volumes, it can be considered zero
- **Price level queue dynamics**
  - Impossible to observe directly

main scope of this research!

# Unobserved Dynamic




Modification



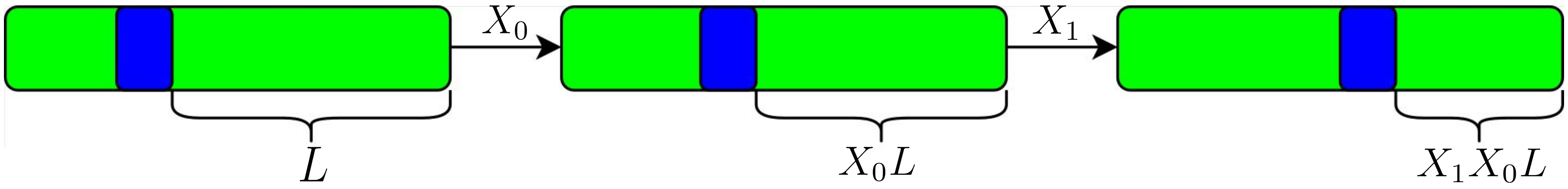

Cancellation

# Problem Statement

Consider sequence of r.v. $(X_t)_{t=1}^{T}$ , where $X_t \sim G(\bullet|O_t)$ bounded with [0, 1].

Here $X_t = \dfrac{L_t}{L_{t-1}}$.

$O_t$ and $L_t$ are order book state and amount before us at timestamp t.




It is necessary to approximate an unknown function using a parameterized model $\hat{G}(\bullet|O_T, \theta)$, where $\theta$ - parameters of the model.

# Aim and Objectives

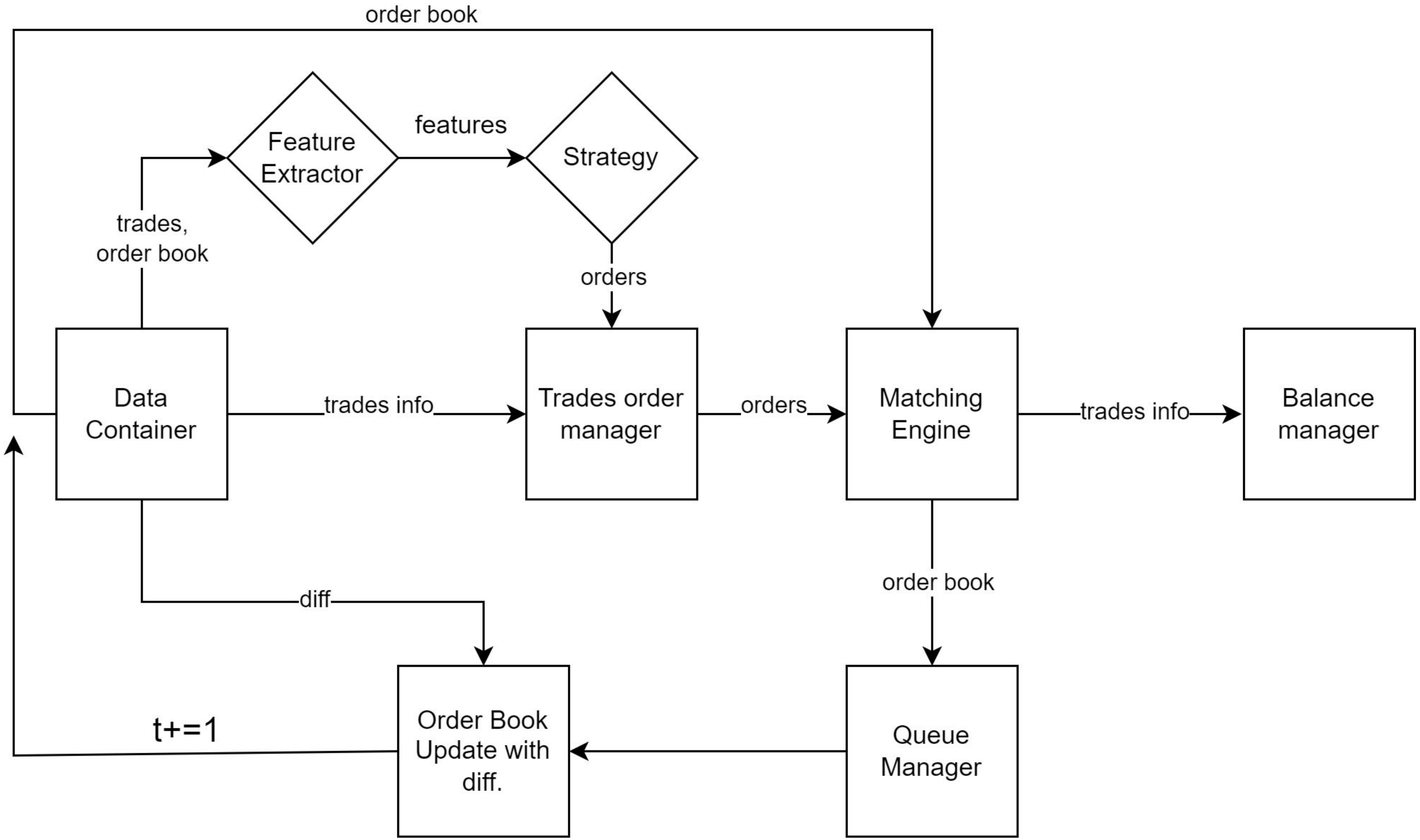
AIM

**Train a reinforcement learning agent to simulate the dynamics of a price level queue**

## Objectives:

- Implement the logic of the order book and the matching engine
- Get exchange data with revealed hidden dynamics
- Design and create an environment for reinforcement learning agent training
- Run experiments and draw conclusions about the ability of the RL agent to learn the dynamics of the price level

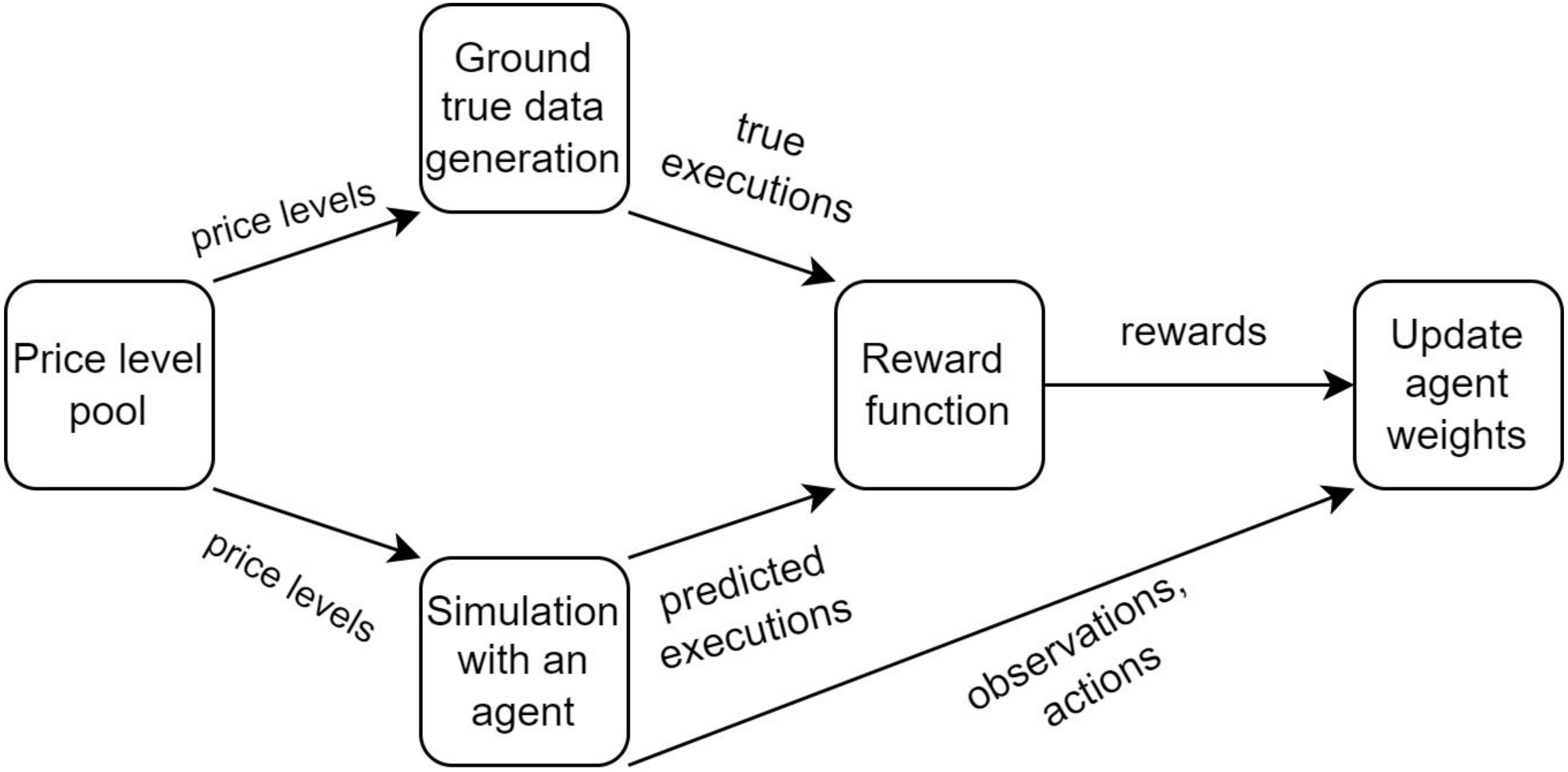
# Backtest


order book
Feature Extractor
features
Strategy
trades,
order book
orders
Data Container
trades info
Trades order manager
orders
Matching Engine
trades info
Balance manager
order book
diff
Order Book Update with diff.
t+=1
Queue Manager

# Environment

- **State**
  - state of the order book, along with its history.
- **Observation**
  - $o_t$ = (number of OB updates, price level volume change, mid-price)
- **Action**
  - $a_t \in [0, 1]$
- **Reward**
  - Sparse reward $r_t = \pm 1$ ; equally distributed among all steps.
- **Step**
  - one update of the order book
- **Episode**
  - lifetime of our limit order in the order book

# Agent training pipeline




Skoltech

# Financial Data

The data was downloaded from the Binance crypto exchange.

30,000+ orderbook updates were used to train the agent.

# Methods

- REINFORCE
    - Ronald J. Williams, “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning”, 1992.
- REINFORCE with baselines
    - Richard S. Sutton, Andrew G. Barto “Reinforcement Learning”
- Actor-Critic
    - Vijay R. Konda, John N. Tsitsiklis, “Actor-Critic Algorithms”, 2000.
- Proximal Policy Optimization
    - J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, “Proximal Policy Optimization Algorithms”, 2017.

# Agent Network

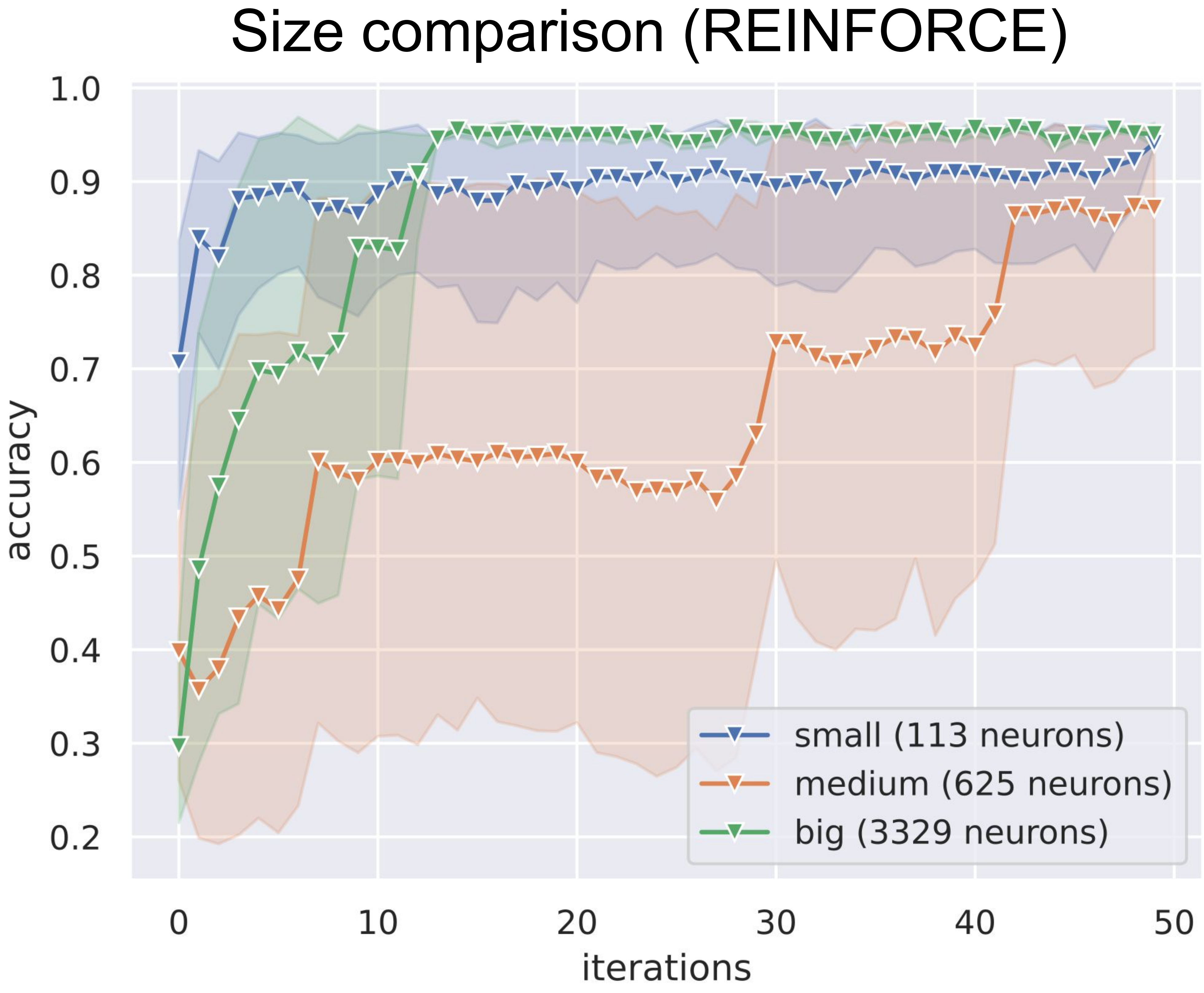
The actions are sampled from the Gaussian distribution

$$\rho^\theta(u \mid y) = \text{pdf}_{\mathcal{N}(\lambda\mu^\theta(y)+\beta, \lambda^2\sigma^2)}(u) = \text{pdf}_{\mathcal{N}(\mu^\theta(y), \sigma^2)}\left(\frac{u-\beta}{\lambda}\right)$$
$$\beta = \frac{u_{\min}+u_{\max}}{2}, \lambda = \frac{u_{\max}-u_{\min}}{2}$$

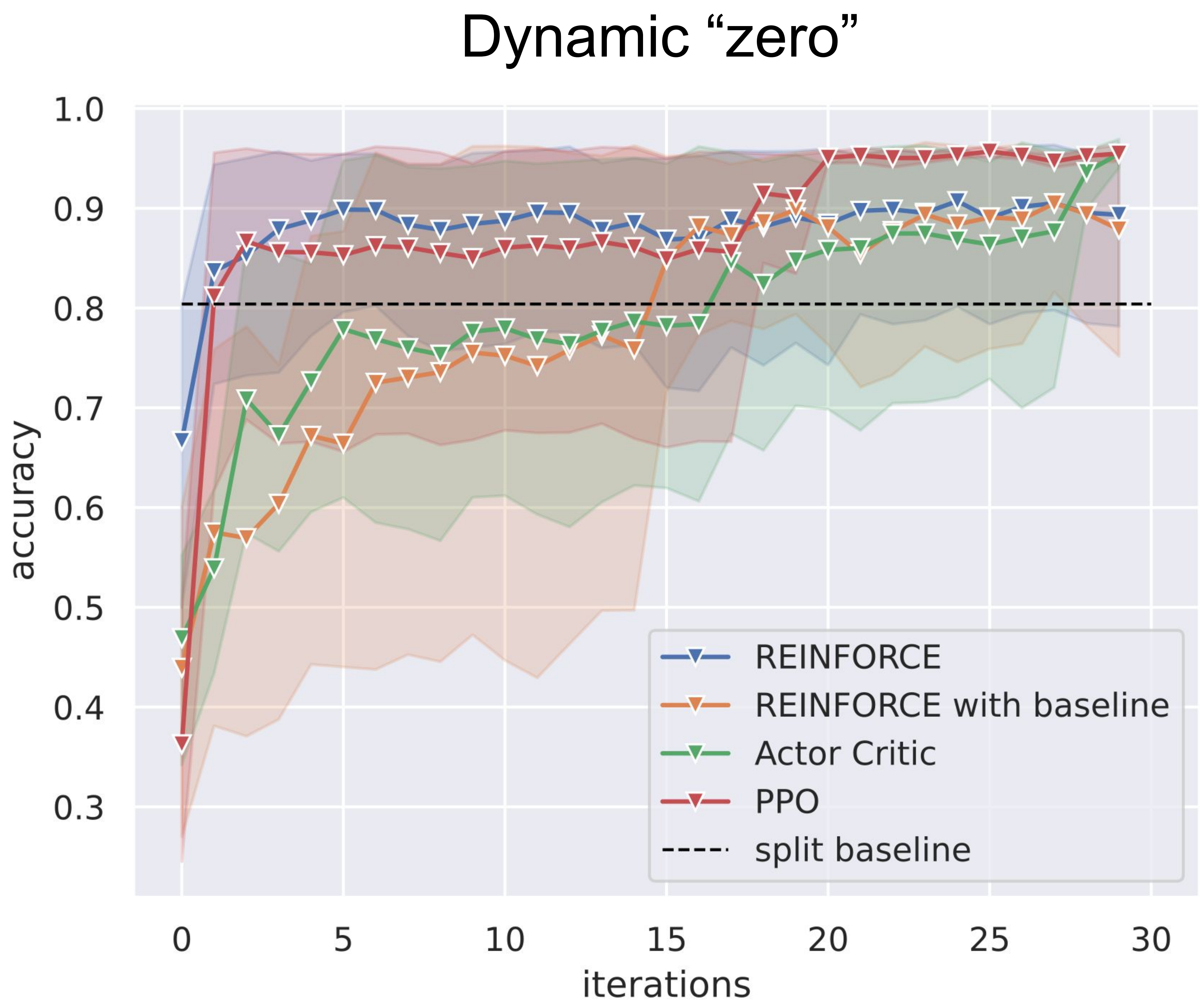
Agent architecture

$$\mu^\theta(y) : y \to \text{Linear}(3,\ ...) \to \text{LeakyReLU} \to \ ... \to \text{Linear}(...,\ 1) \to (1 - 3\sigma)\tanh\left(\frac{\cdot}{L}\right)$$

# Results

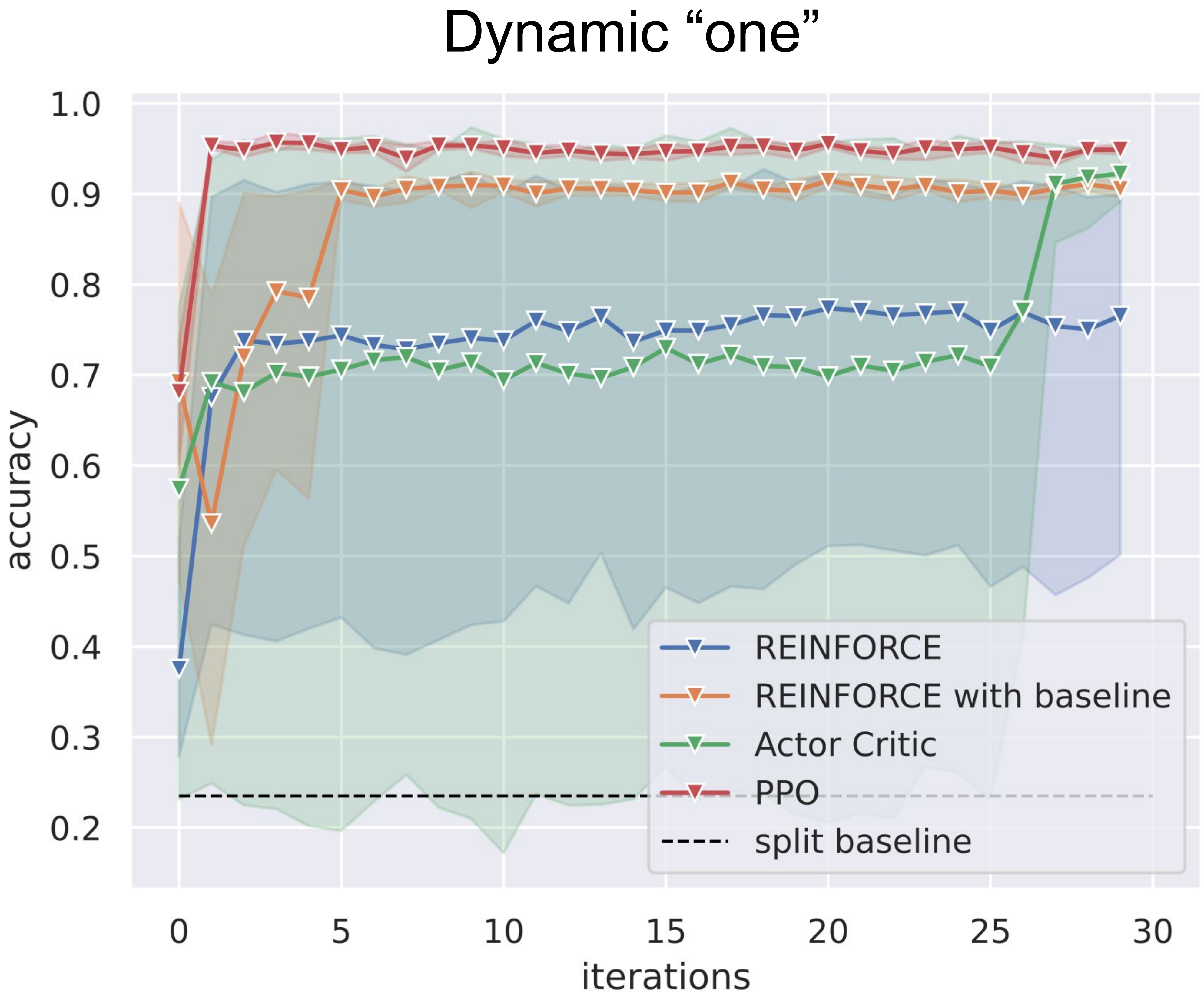

Size comparison (REINFORCE)
accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
10
20
30
40
50
iterations
small (113 neurons)
medium (625 neurons)
big (3329 neurons)

# Results



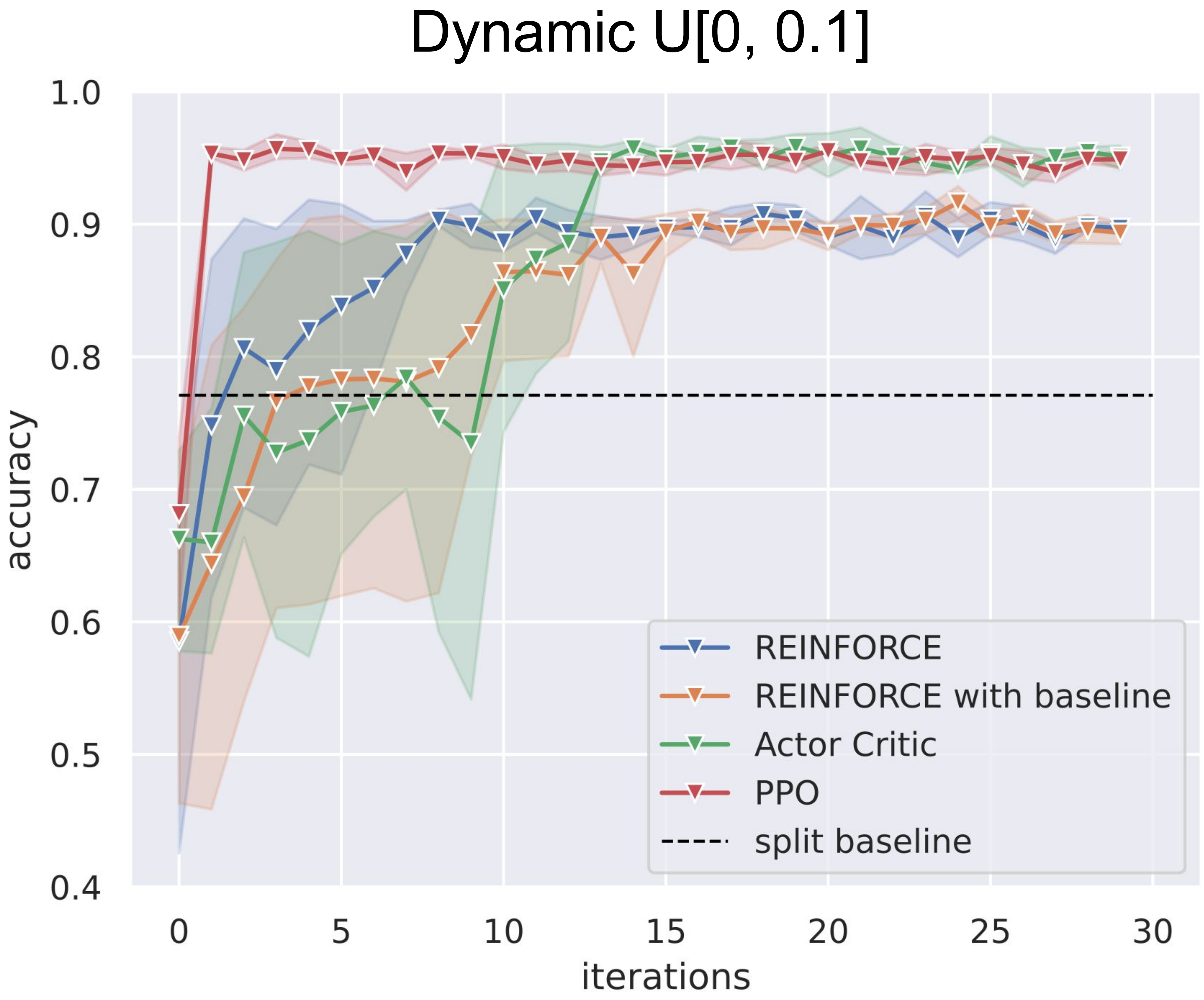

Skoltech

# Results



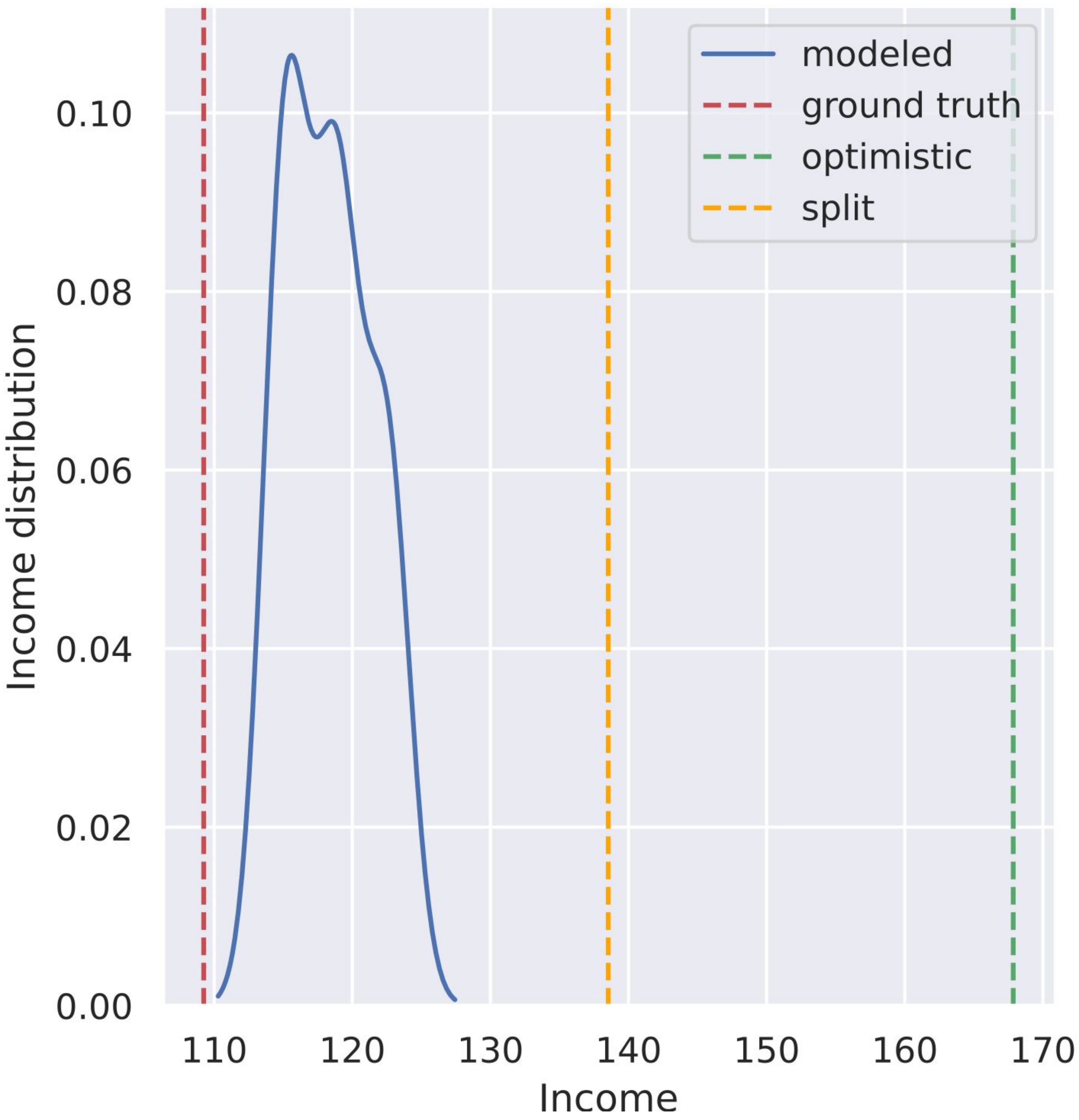

Skoltech

# Results



Dynamic U[0, 0.1]
accuracy
iterations
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
5
10
15
20
25
30
REINFORCE
REINFORCE with baseline
Actor Critic
PPO
split baseline

# Results




Comparison of strategy evaluation

Skoltech

# Discussion of results

**Outcomes:**

- The RL algorithms proved to be able to successfully simulate the given dynamics
- The larger the size of the agent's model, the longer its training takes to converge. But as a result, the reward curves are more stable.
- The PPO method showed the best results. In his case, the agent learns faster and more stable than other methods.

**Limitations:**

- The dynamics of the queues of price levels was synthetically generated.

# Conclusions

- Code has been written to implement the logic of an orderbook with a latent queue.
- A pipeline was invented and implemented to train RL agents.
- Observation, action and reward engineering performed.
- Reinforcement learning methods have shown the ability to learn the hidden dynamics of the order book queue.

# Scientific novelty

- A study of the dynamics of the microstructure of the market-by-level exchange.
- Modeling the dynamics of the order book using reinforcement learning methods.

# Acknowledgements


- Thanks to Georgiy Malaniya, a PhD student at the AIDA lab, for his enthusiastic help with ideas, help with immersion in the subject area and mentoring.
- And many thanks to my supervisor Pavel Osinenko for his advice and help in writing the thesis.

# Thx

Skoltech

# Appendix. Backtest

**Data:** Initial OB state, trades and OB updates.

**Input:** $S$ - strategy, $T$ - last timestamp, $ME$ - matching engine, $OM$ - order manager, $QM$ - queue manager.

Initialize $OB_0$ - order book at timestamp 0 with start OB state.

t = 0

**while** $t < T$ **do**

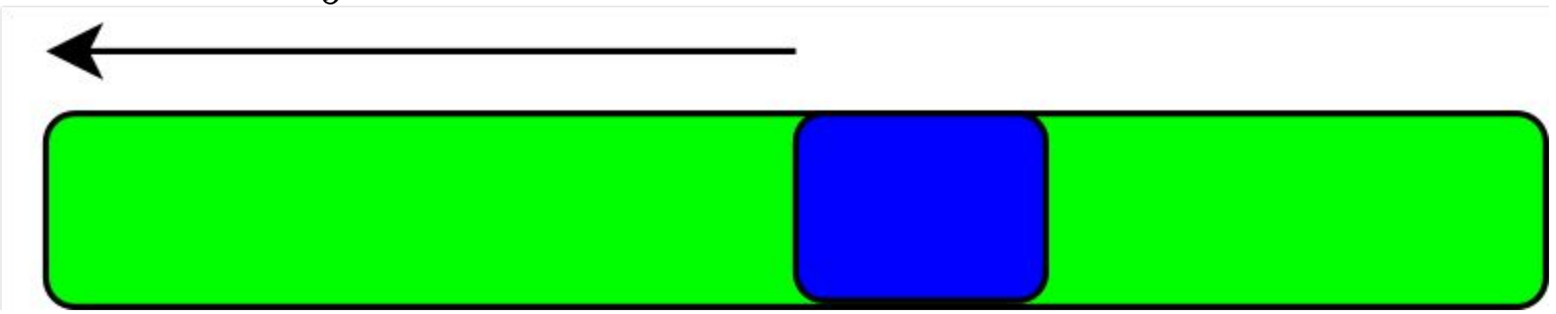
- Sample trades $Y_t$ that happened between t and t+1 and $D_t$ - OB update with a diff.
- $S(OB_t, Y_t) \rightarrow Y_T^S$ - strategy generate orders based on OB and trades info.
- $OM(Y_t, Y_t^S) \rightarrow Y_t^{ordered}$ - order manager combines historical market orders with strategy's orders.
- $Y_t^{ordered} \rightarrow OB_t$ - update OB with orders.
- $QM(OB_t)$ - update strategy's orders positions.
- $D_t \rightarrow OB_t$ - update OB with the diff.
- t = t + 1

**end**

# Appendix. Queue dynamic generation

- Limit order size is 1
- Price levels are not executed in one trade
- Assign historical liquidity as ours
- Simulation of stochasticity
  - no queue dynamic, or sampling zeros
  - push to the front, or sampling ones
  - small fluctuation, or sampling from U[0, 0.1]

$$\Delta l = f\Delta t + \sigma \Delta W$$



Skoltech

# Appendix. Gradient methods in RL

Consider system:

$$X_{k+1} \sim \hat{f}(x_{k+1} \mid x_k, u_k), \quad Y_k = h(X_k) \sim f(h(x_k) \mid x_{k-1}, u_{k-1}), \quad U_k \sim \rho^\theta(u_k \mid y_k).$$

Objective function:

$$\max_\theta J_N(\theta) = \mathbb{E}_{f,\rho^\theta} \left( \sum_{k=0}^{N-1} \gamma^k r(Y_k, U_k) \right)$$

# Appendix. Methods

REINFORCE

$$\theta_{i+1} = \theta_i + \alpha_i \frac{1}{M} \sum_{j=1}^{M} \left( \sum_{k=0}^{N_j-1} \sum_{l=k}^{N_j-1} \gamma^l r(y_l^j, u_l^j) \nabla_\theta \ln \rho^\theta(u_k^j \mid y_k^j)\big|_{\theta=\theta_i} \right)$$

REINFORCE with baselines

$$\theta_{i+1} = \theta_i + \alpha_i \frac{1}{M} \sum_{j=1}^{M} (\sum_{k=0}^{N_j-1} (\sum_{l=k}^{N_j-1} \gamma^l r(y_l^j, u_l^j) - B_k) \nabla_\theta \ln \rho^\theta(u_k^j \mid y_k^j)\big|_{\theta=\theta^i})$$

baseline formula:

$$B_k = \frac{1}{M} \sum_{j=1}^{M} \sum_{k'=k}^{N_j-1} \gamma^{k'} r(y_{k'}^j, u_{k'}^j)$$

# Appendix. Methods

## Actor

$$\theta_{i+1} = \theta_i + \alpha_i \frac{1}{M} \sum_{j=1}^{M} \left( \sum_{k=0}^{N_j-2} \gamma^k \left( r(y_k^j, u_k^j) + \gamma \hat{J}^w(y_{k+1}^j) - \hat{J}^w(y_k^j) \right) \nabla_\theta \ln \rho^\theta(u_k^j \mid y_k^j) \Big|_{\theta=\theta_i} \right)$$

## Critic

$$w^{new} = w^{old} - \eta \nabla_w \left[ \frac{\sum\limits_{k=0}^{N_j-1-N_{\mathrm{TD}}} (\hat{J}^w(y_k^j) - r(y_k^j, u_k^j) - \gamma r(y_{k+1}^j, u_{k+1}^j) - \ldots - \gamma^{N_{\mathrm{TD}}} \hat{J}^w(y_{k+N_{\mathrm{TD}}}^j))^2}{N_j - 1 - N_{\mathrm{TD}}} \right] \Bigg|_{w=w^{old}}$$

## Proximal Policy Optimization

$$\theta^{\mathrm{new}} = \theta^{\mathrm{old}} + \alpha \nabla_\theta \left( \frac{1}{M} \sum_{j=1}^{M} \sum_{k=0}^{N_j-2} \gamma^k \max \left( \hat{A}^w(y_k^j, u_k^j) \frac{\rho^\theta(u_k^j | y_k^j)}{\rho^{\theta_i}(u_k^j | y_k^j)}, \hat{A}^w(y_k^j, u_k^j) \, \mathrm{clip}_{1-\varepsilon}^{1+\varepsilon} \left( \frac{\rho^\theta(u_k^j | y_k^j)}{\rho^{\theta_i}(u_k^j | y_k^j)} \right) \right) \right) \Bigg|_{\theta=\theta^{\mathrm{old}}}$$

$$\hat{A}^w(y_k^j, u_k^j) = r(y_k^j, u_k^j) + \gamma \hat{J}^w(y_{k+1}^j) - \hat{J}^w(y_k^j)$$