ETH zürich

Deep Reinforcement Learning for Double Auction Processes

Batuhan Yardim

Aleksei Khudorozhkov

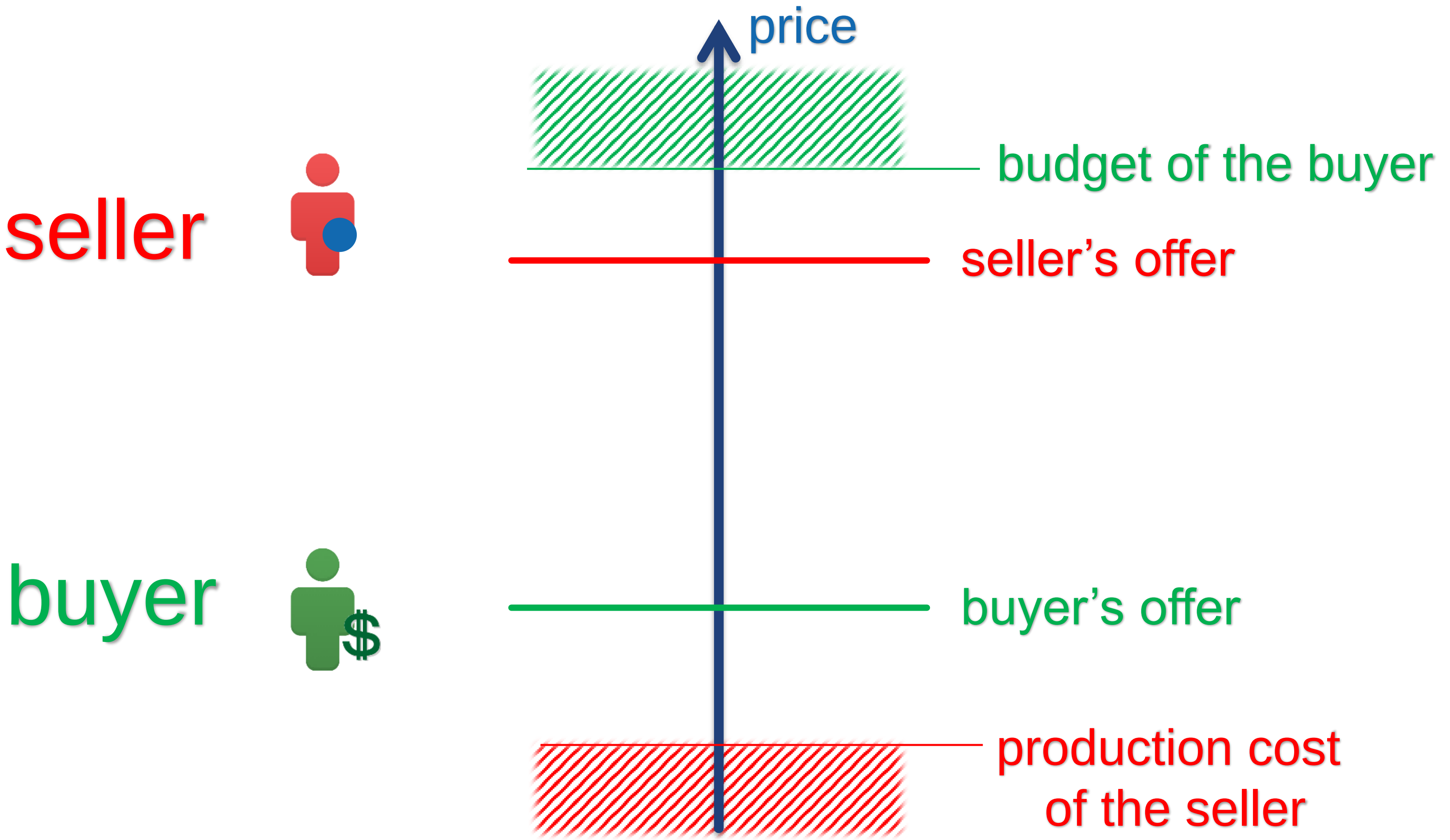
Ka Rin Sim

Neri Passaleva

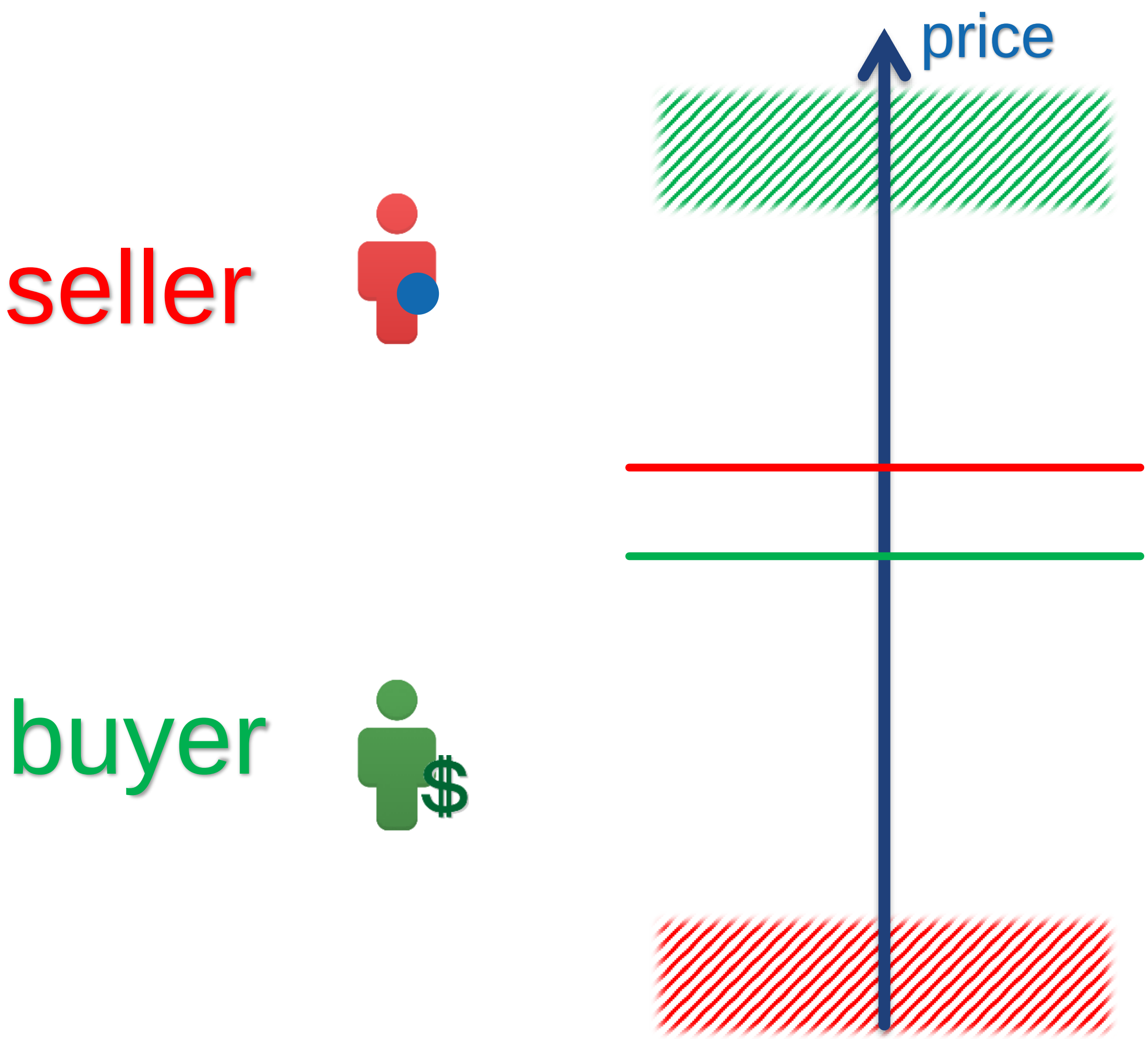
Goals

- Simulate double auction processes
- Intelligent & non-intelligent agents
- Implement reinforcement learning for intelligent agents

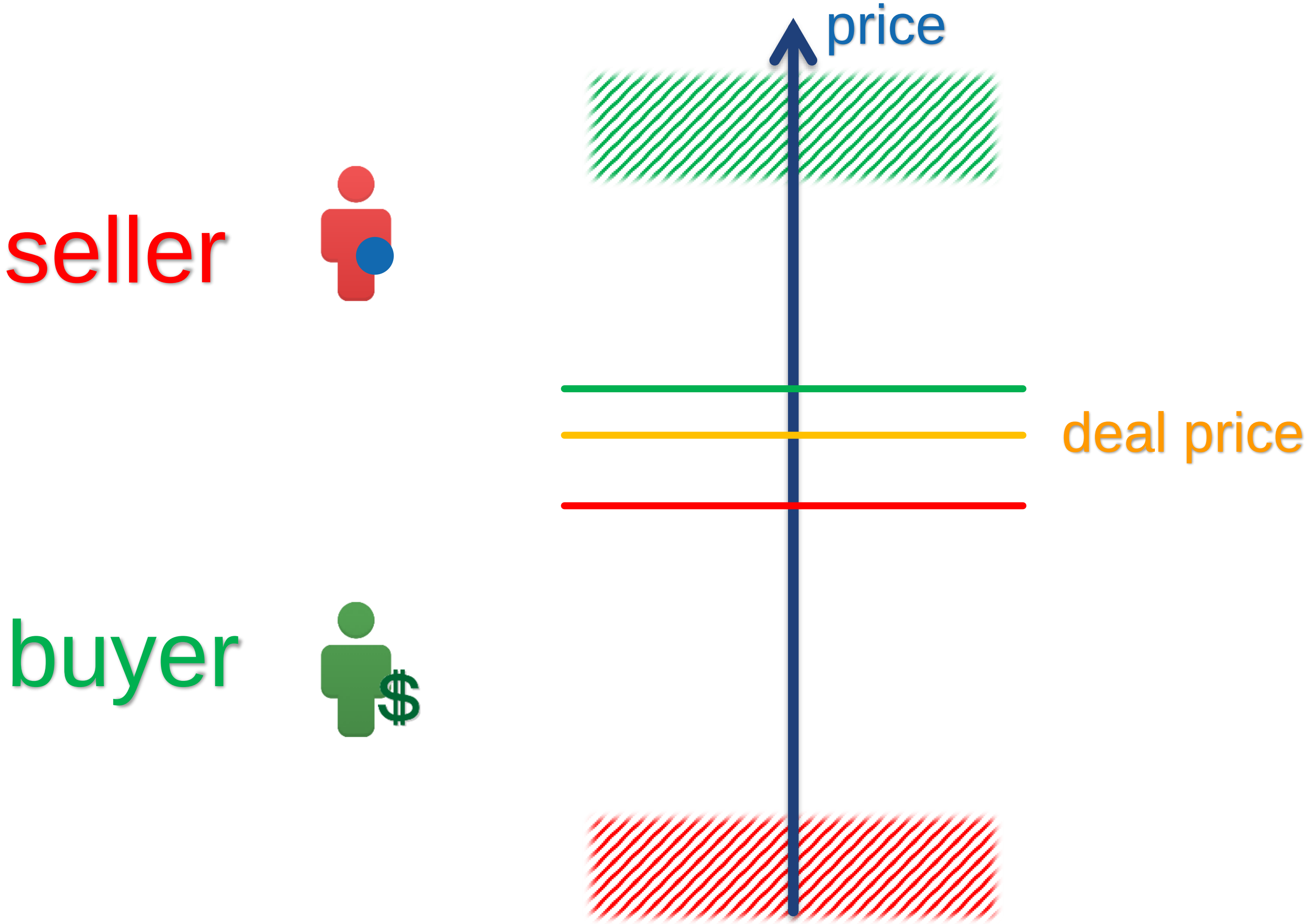
Double auction



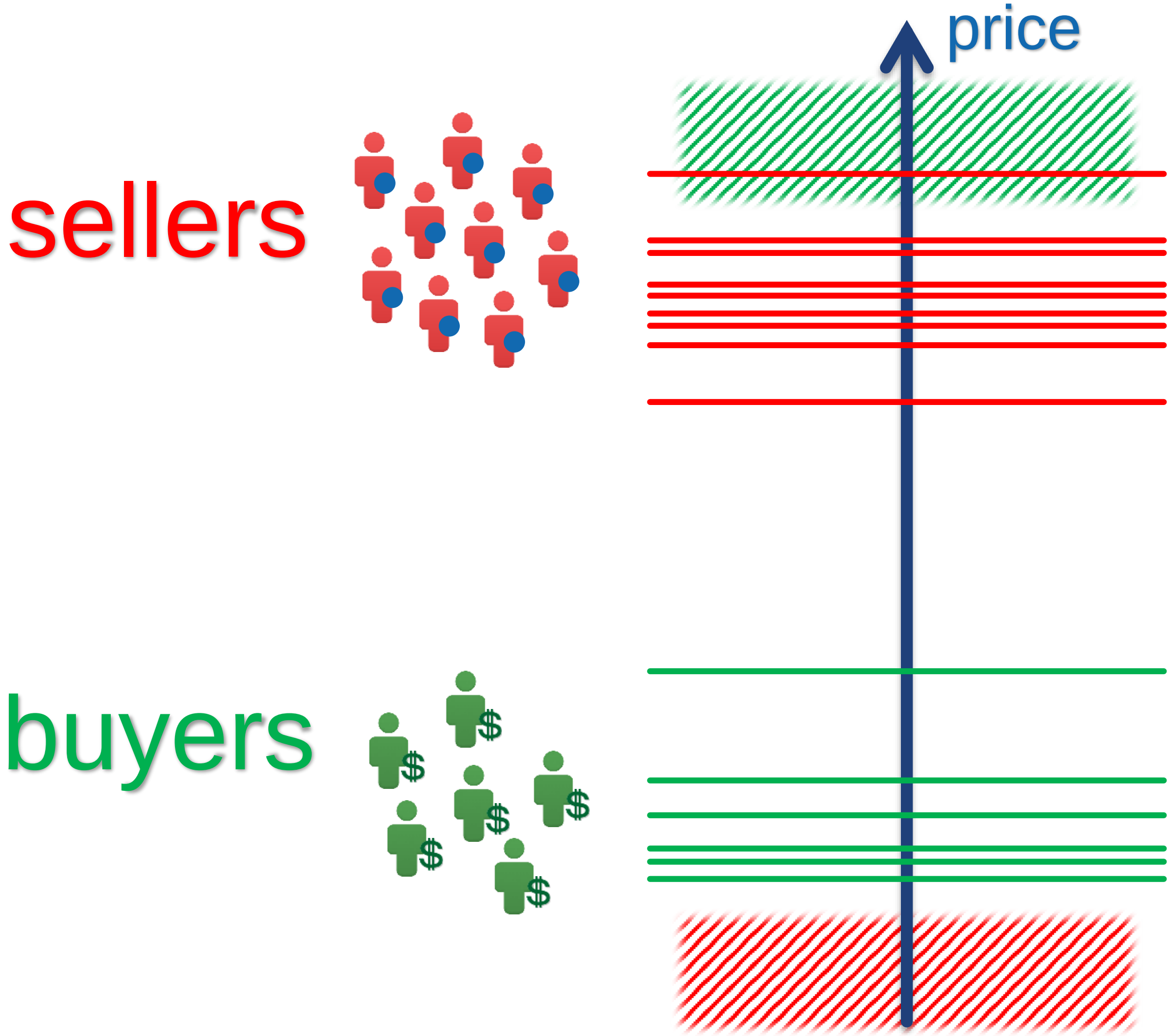
Double auction



Double auction



Double auction

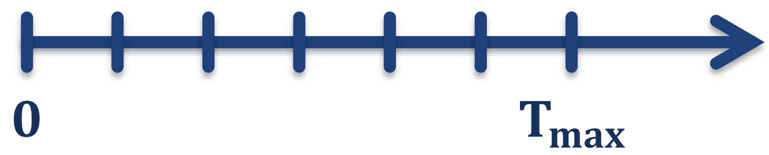


Each agent wants to maximize the reward

For this they can choose different strategies

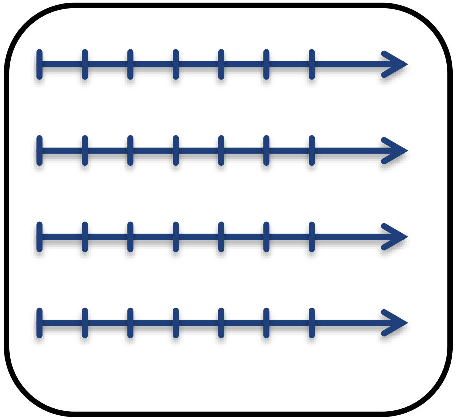
Market environment

- Each round consists of time steps



  - Round terminates when T_{max} is reached or no more deals can be made

- Each game consists of rounds



  - Agents can have memory about the previous rounds
  - Between the games agents can learn and adjust their strategy

Observations of agents

- After each time step an agent receives observations from the market environment.

- Core observations are:
  - The last offer of the agent
  - Current time step
  - if the agent managed to make a deal in the previous round

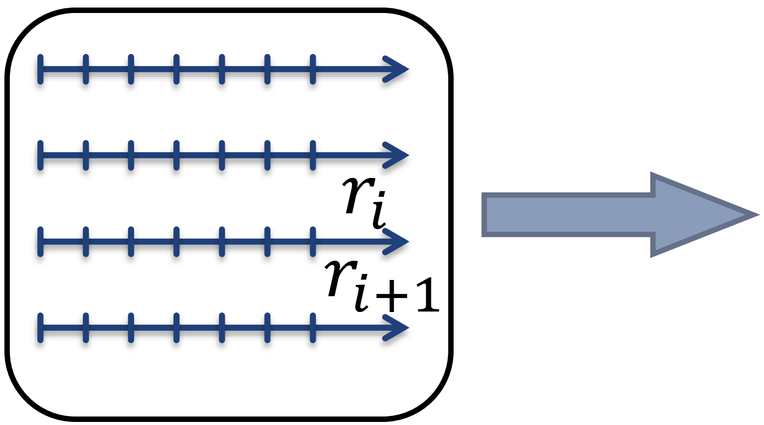
- Other observations might be included.

Reward Mechanism

- The agent’s **reward** is the absolute difference between the reservation price and the agent’s deal offer.

$$r_i = |p_i - a_i|$$ reward for round *i*

- Reward is cumulative throughout the rounds



$$r = \sum_i r_i$$ total reward

Zero-Intelligence Agent (ZIA)

The agent randomly chooses the next offer according to the exponential distribution around the reservation price

No observations are needed in order to decide on a new offer

Linear Markov Decision Agent (LMDA)

- The new demand is a linear combination of the agent's observations

Demand at previous time step

Demand at current time step

$$d_i = \alpha d_{i-1} + \beta s + n_i$$

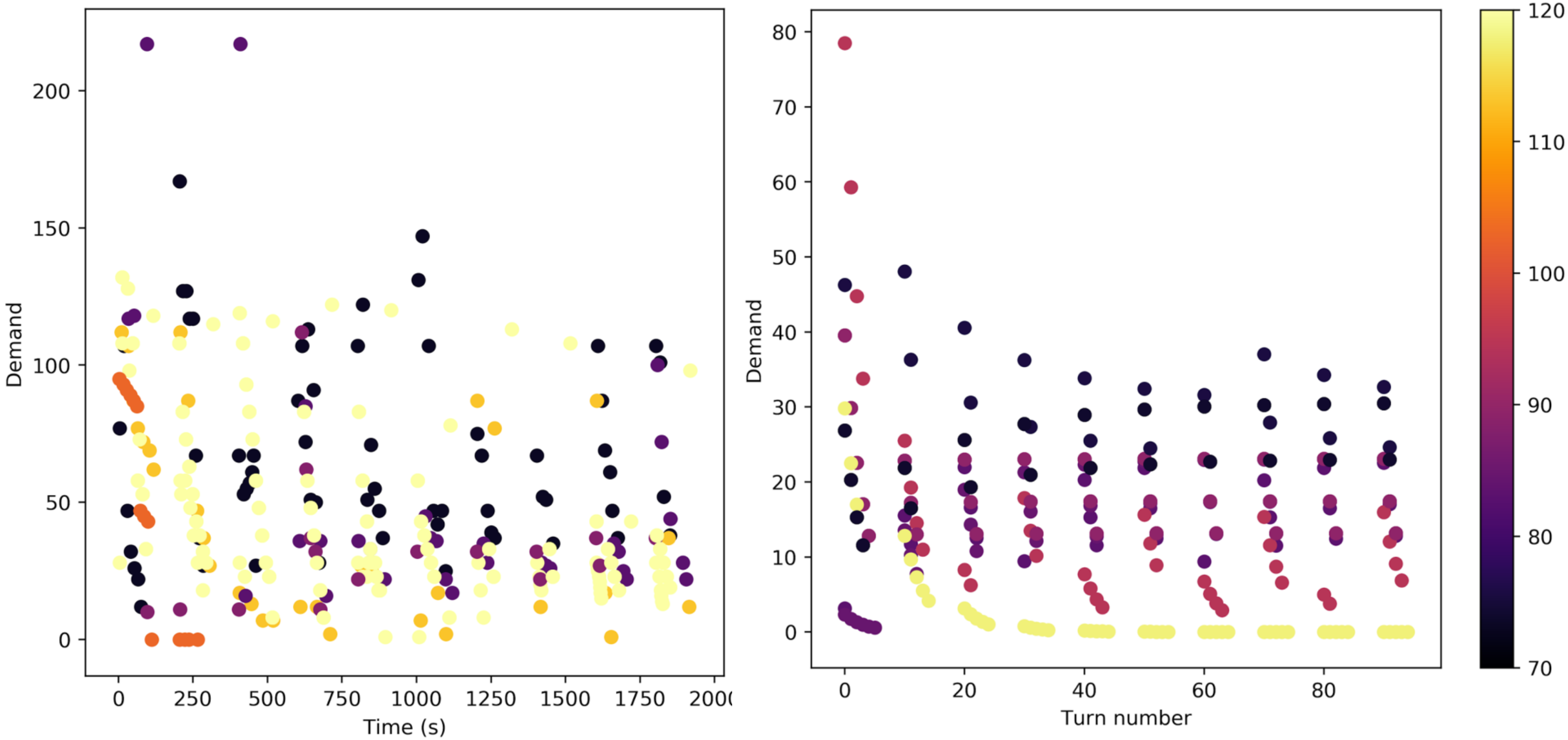
Boolean indicator of previous round outcome

Noise

$$s = \begin{cases} 0 & if\ unsuccessful \\ 1 & if\ successful \end{cases}$$

Results: LMDA vs real people

- LMDA is a first-order approximation of real people



Real people experiments

LMDA (no learning)

Price Aggressive Agents (PAA)

s is used as an indicator for whether the agent should be aggressive or not

If ***s*** is **TRUE:**

$$d_i = \alpha d_{i-1} + n_i$$

If ***s*** is **FALSE**

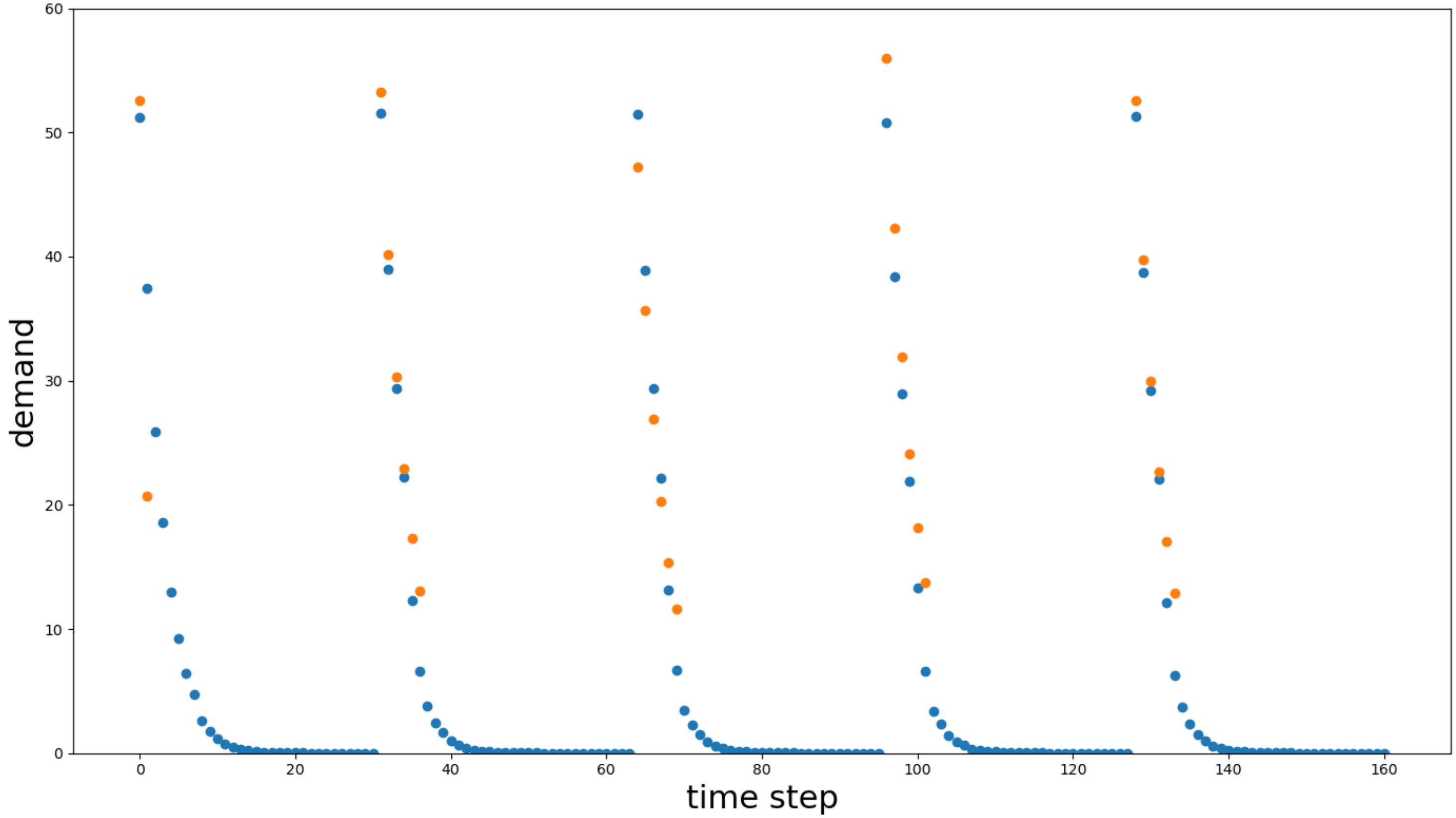
$$d_i = (\alpha + \varepsilon) d_{i-1} + n_i$$

ε is the agent's level of aggressiveness

The agent becomes aggressive after an unsuccessful round and tries to make a deal even with low reward

Results: one PAA among LMDA

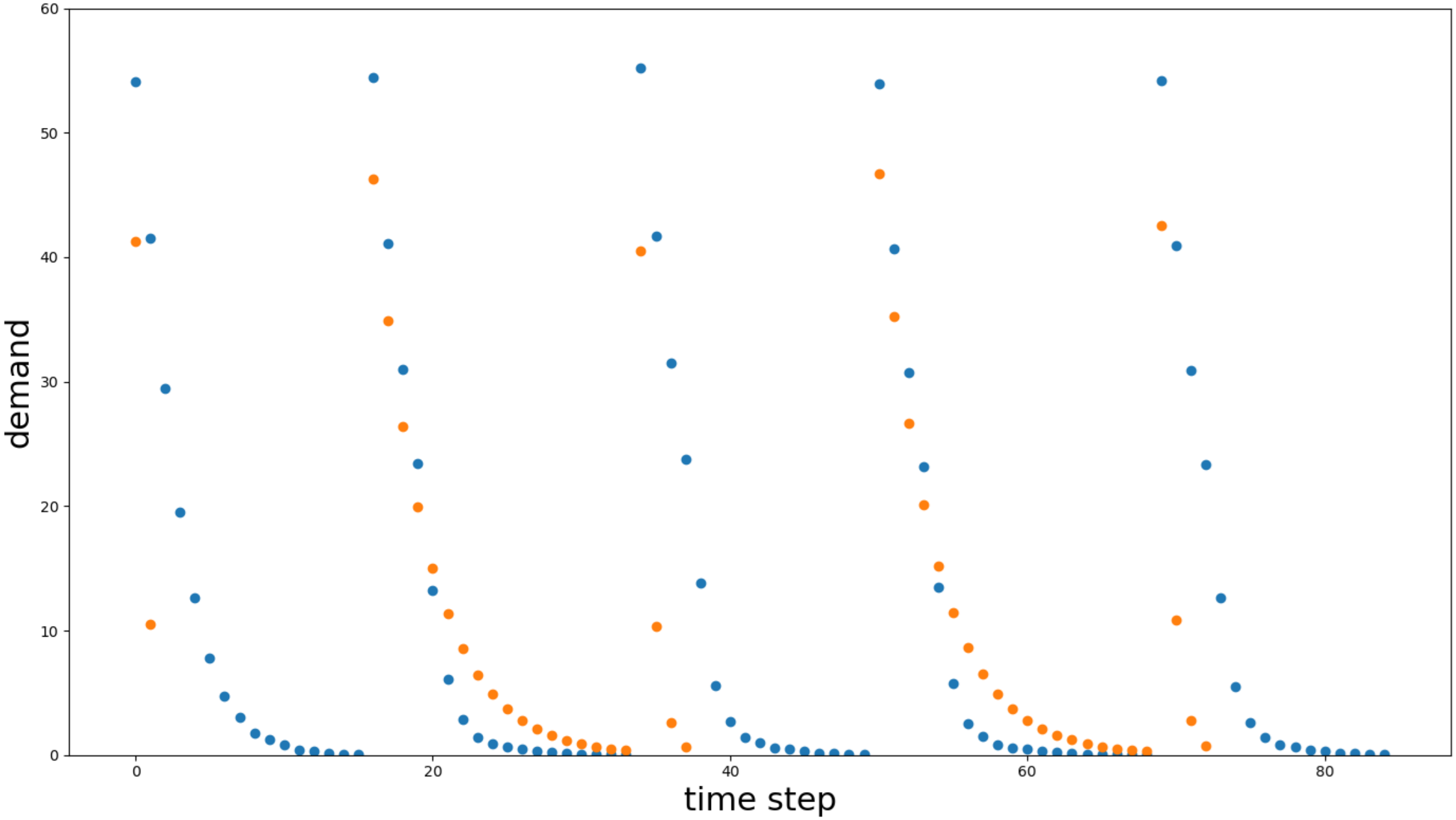
- PAA with zero aggressiveness is equal to LMDA



- **average demand of LMDA agents**
- **demand of one PAA agent**

Results: one PAA among LMDA

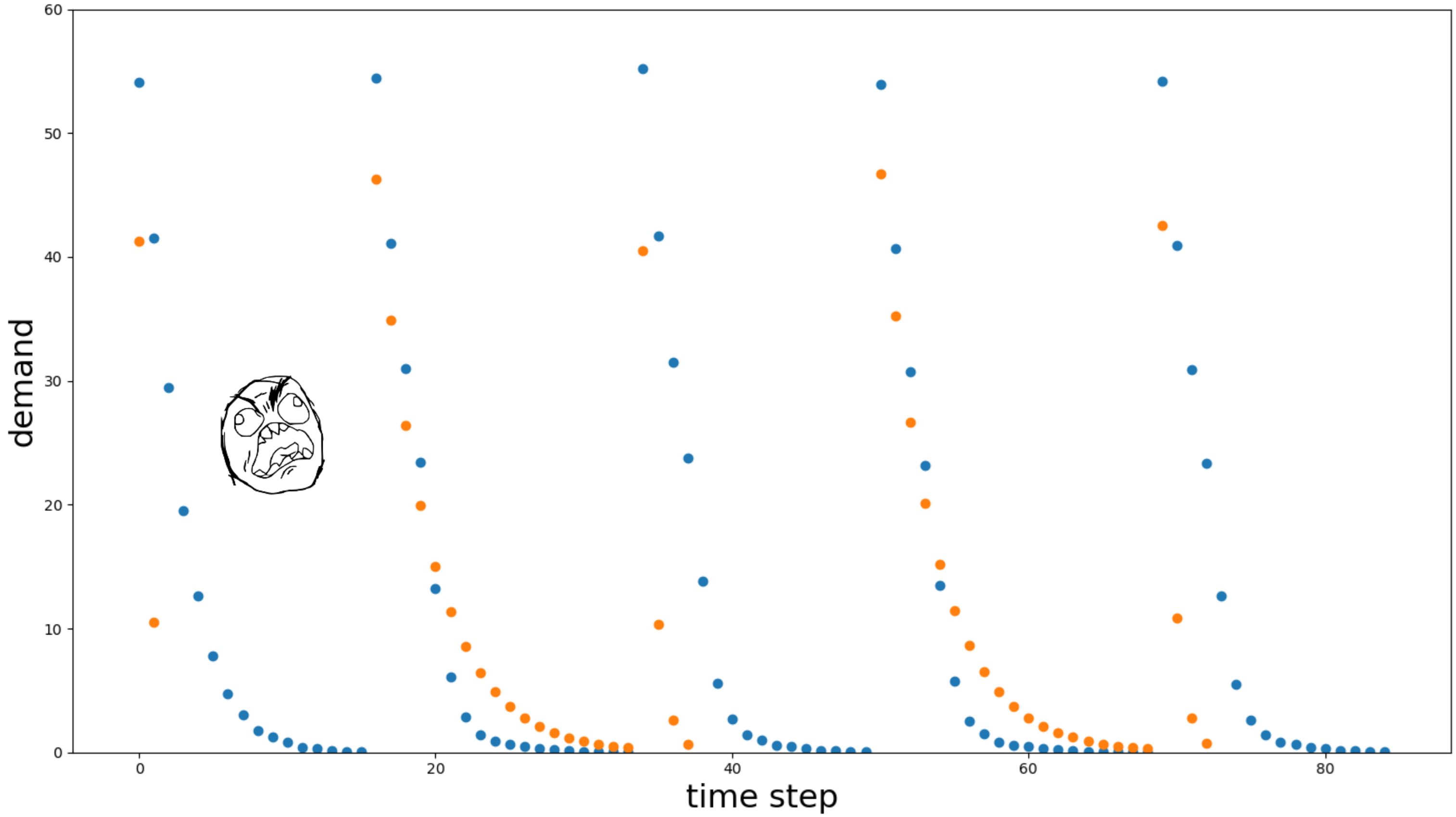
- PAA with aggressiveness $\varepsilon = 0.5$



- average demand of LMDA agents
- demand of one PAA agent

Results: one PAA among LMDA

- PAA with aggressiveness $\varepsilon = 0.5$

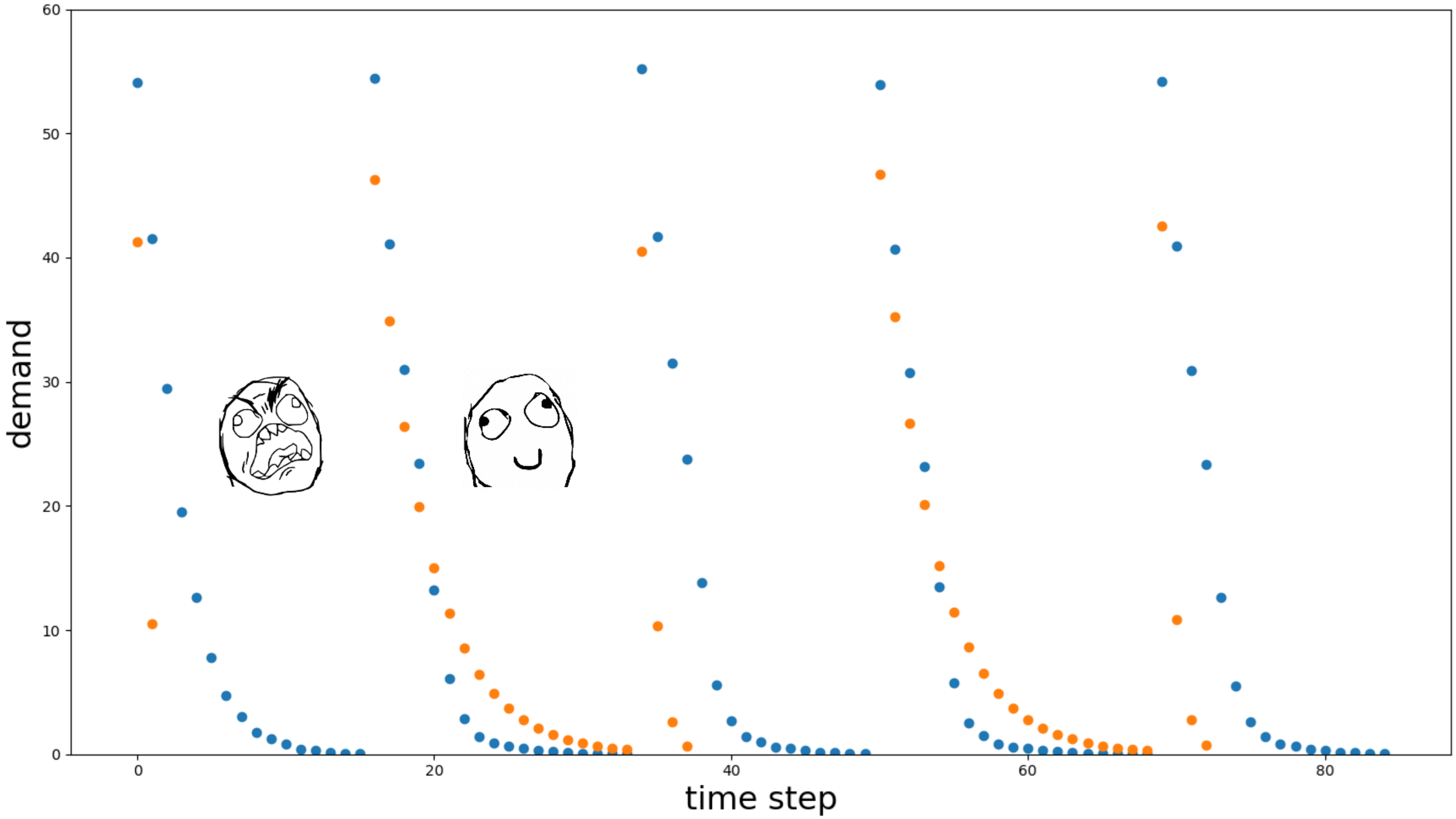


● **average demand of LMDA agents**

● **demand of one PAA agent**

Results: one PAA among LMDA

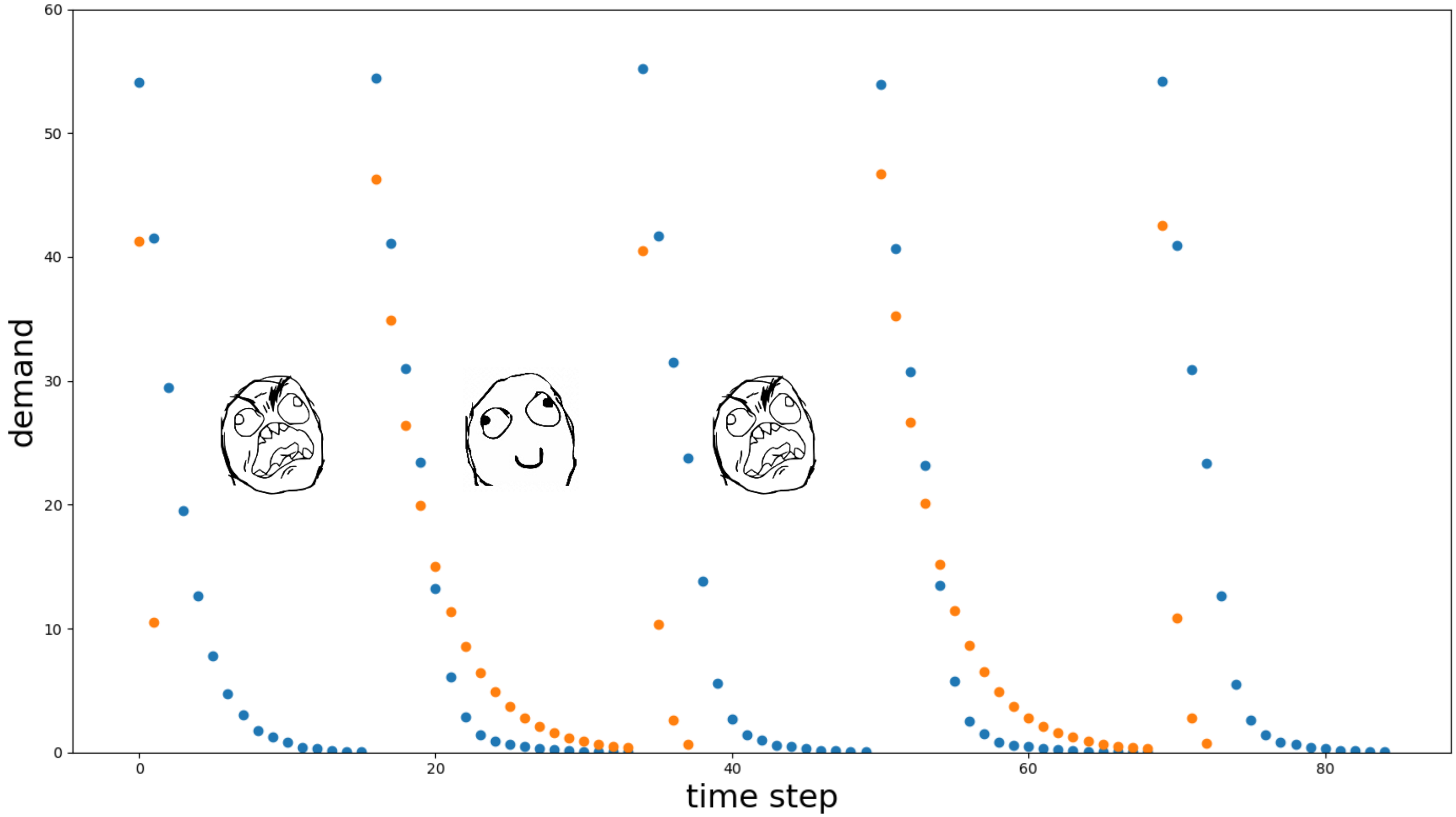
- PAA with aggressiveness $\varepsilon = 0.5$



- **average demand of LMDA agents**
- **demand of one PAA agent**

Results: one PAA among LMDA

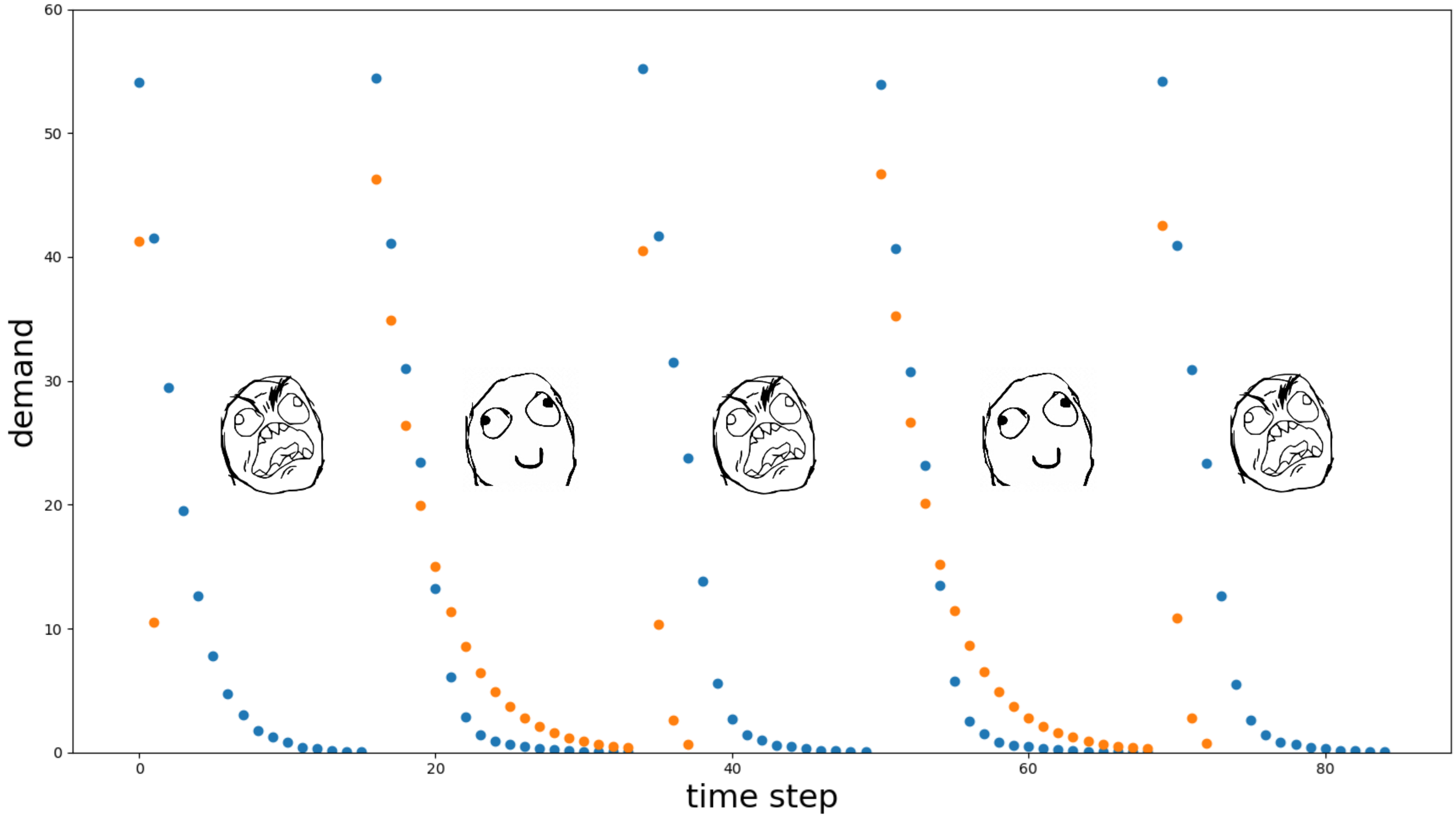
- PAA with aggressiveness $\varepsilon = 0.5$



- **average demand of LMDA agents**
- **demand of one PAA agent**

Results: one PAA among LMDA

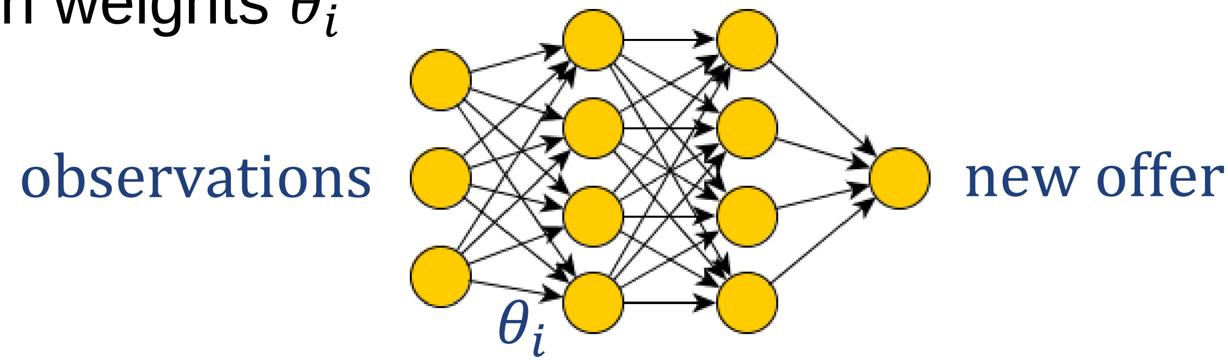
- PAA with aggressiveness $\varepsilon = 0.5$



- **average demand of LMDA agents**
- **demand of one PAA agent**

Deep RL Agents

- The new offer decision mechanism is a neural network with weights θ_i



$$d_i = \pi_{\theta_i}(o_i) + \mathcal{N}(0, \sigma_i)$$

Parametrized Agent's Policy

Gaussian noise

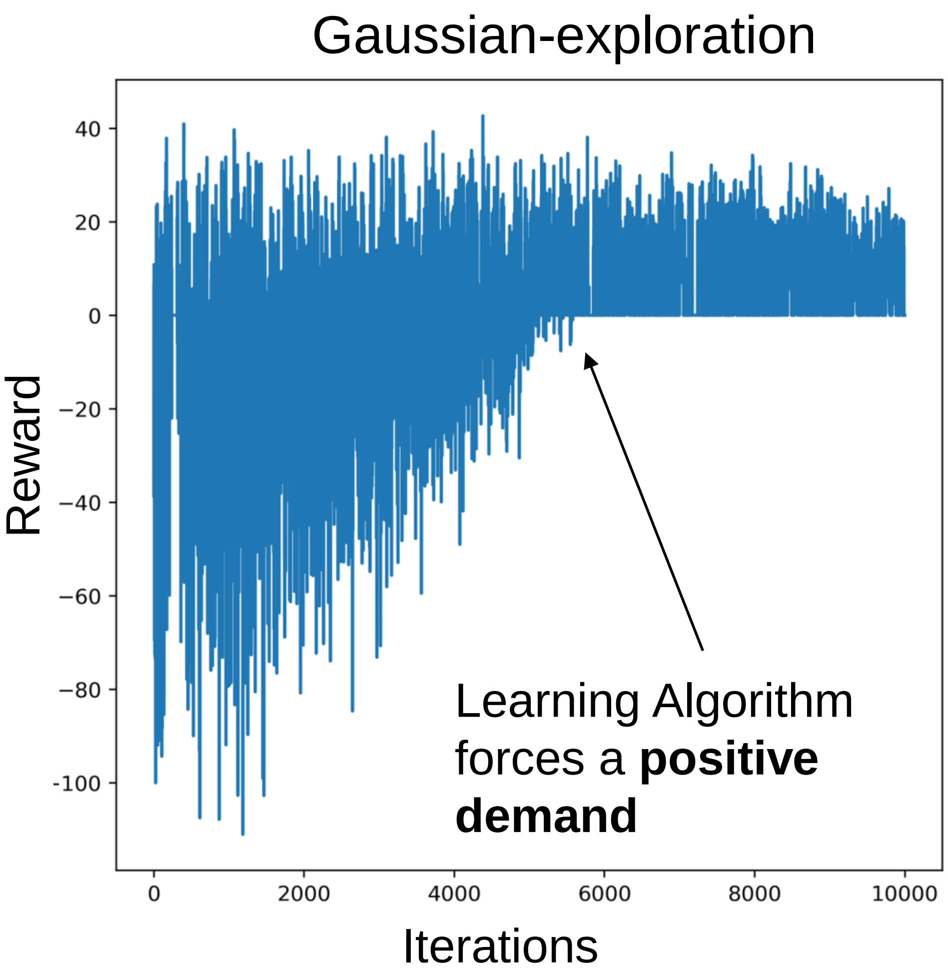
- Reinforcement learning through Deep Deterministic Policy Gradient (DDPG) framework

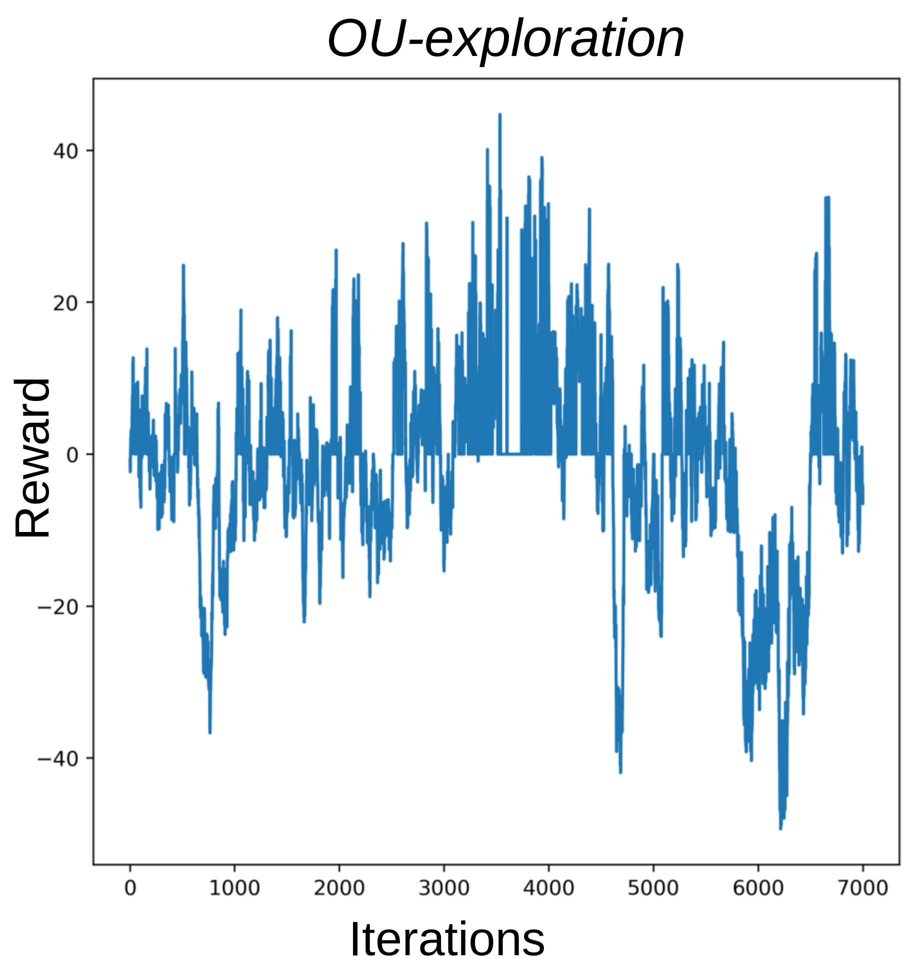
Deep RL Agents

- 2 different exploration policies:
  - Gaussian
  - Ornstein-Uhlenbeck (OU)

- 1 intelligent agent + a pool of non-intelligent agents (e.g. ZIA, LMDA)
- No structural assumptions about game

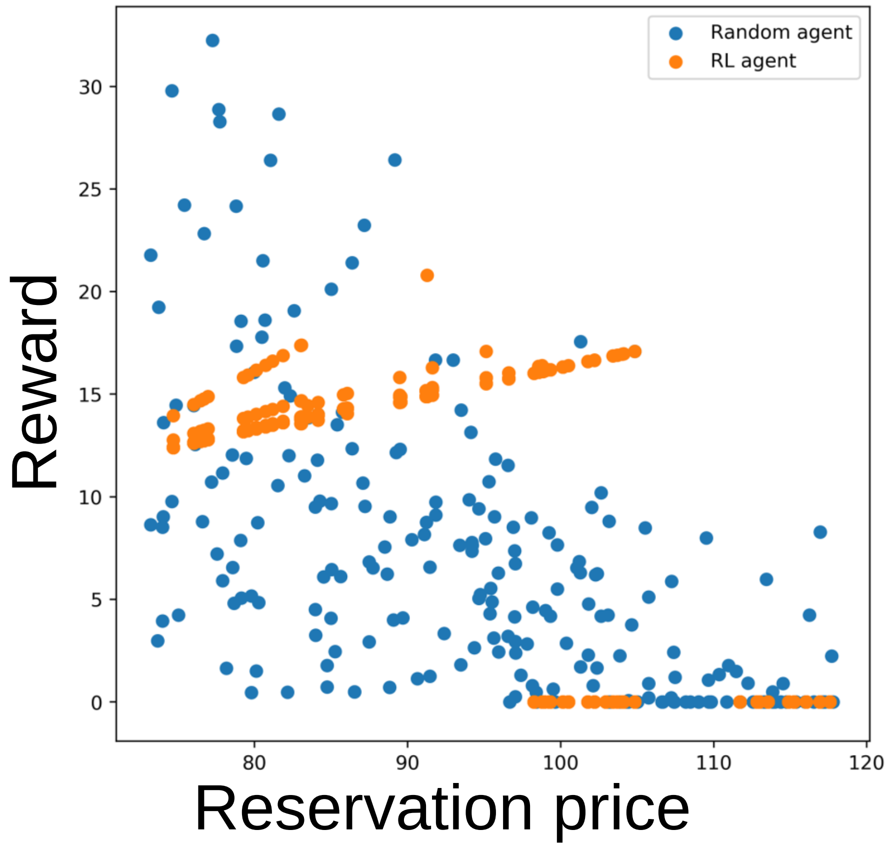
Effect of Exploration Policy



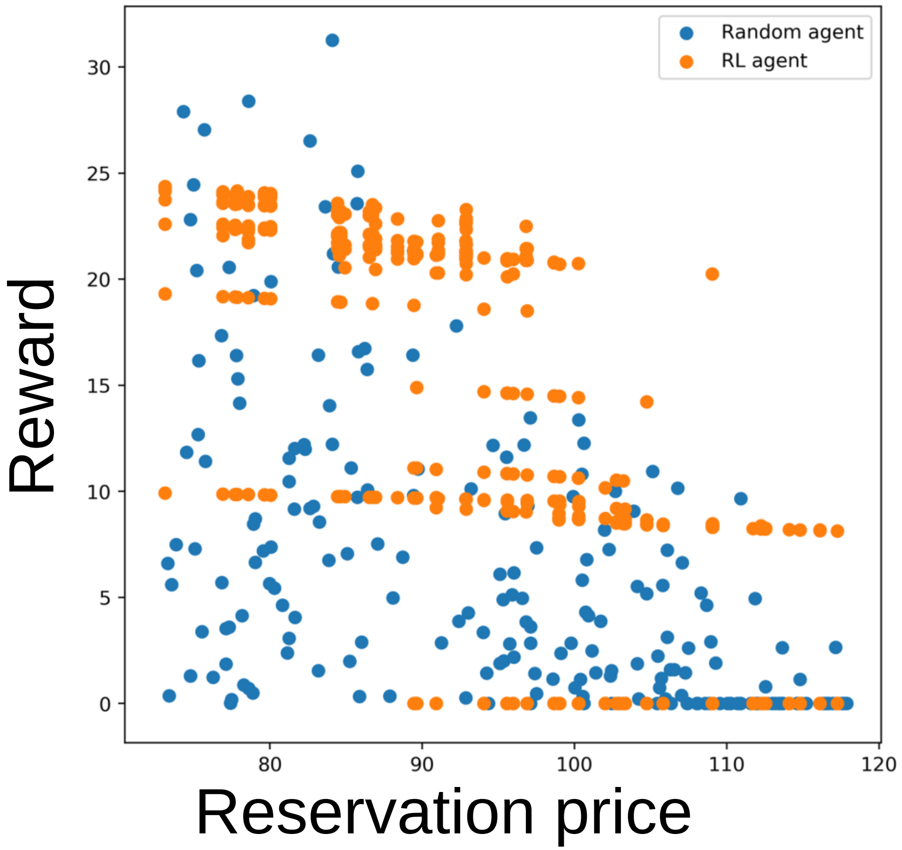


Deep RL Agent vs ZIA

Gaussian-exploration



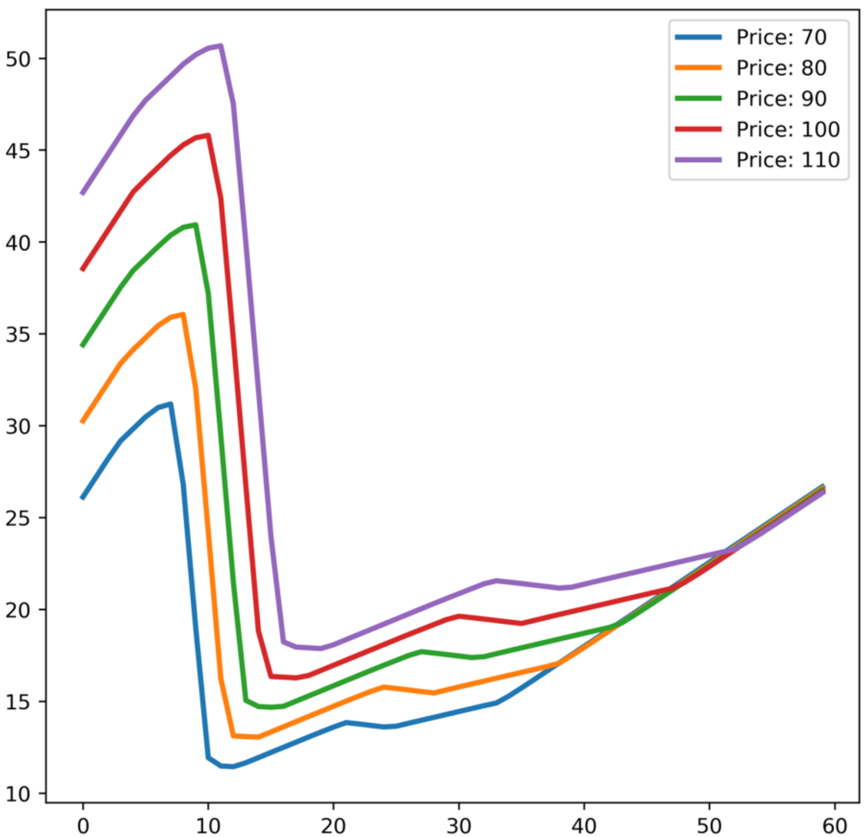
OU-exploration



Deep RL Agent

Gaussian-exploration

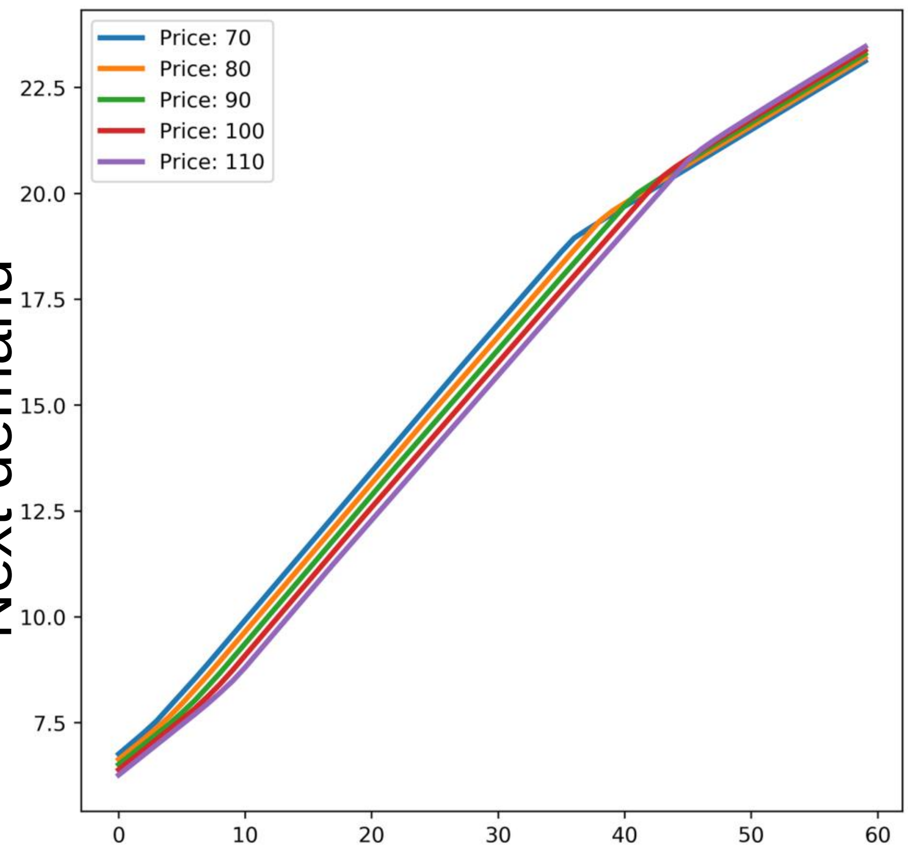
Next demand



Last unsuccessful demand

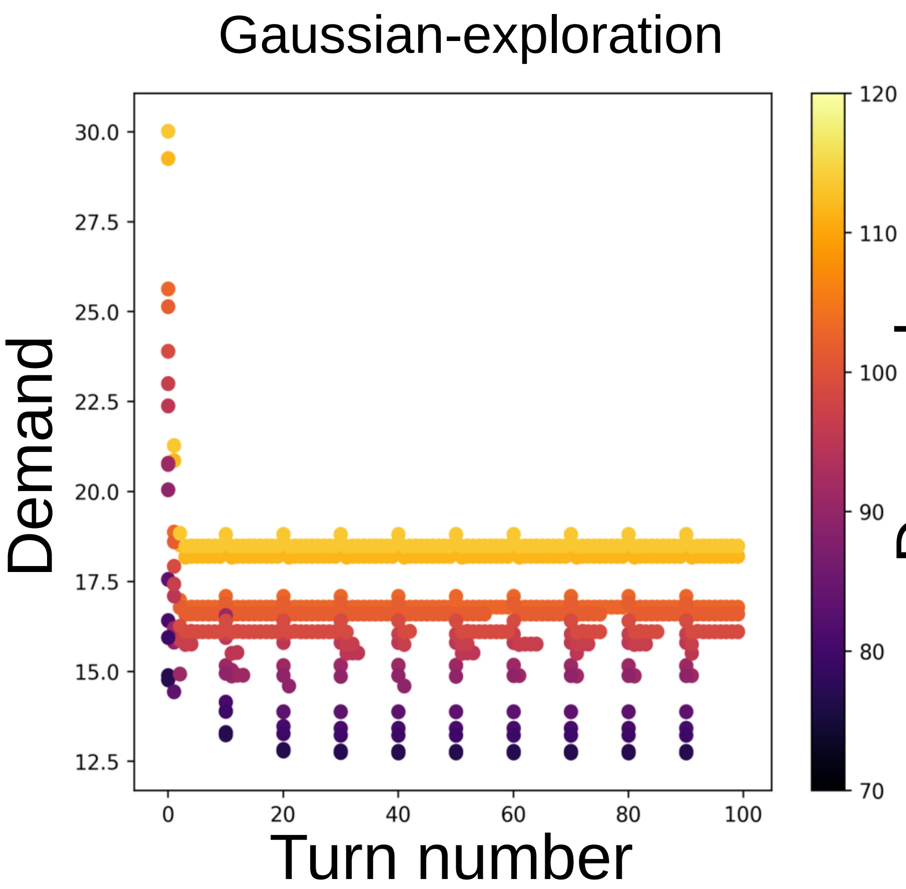
OU-exploration

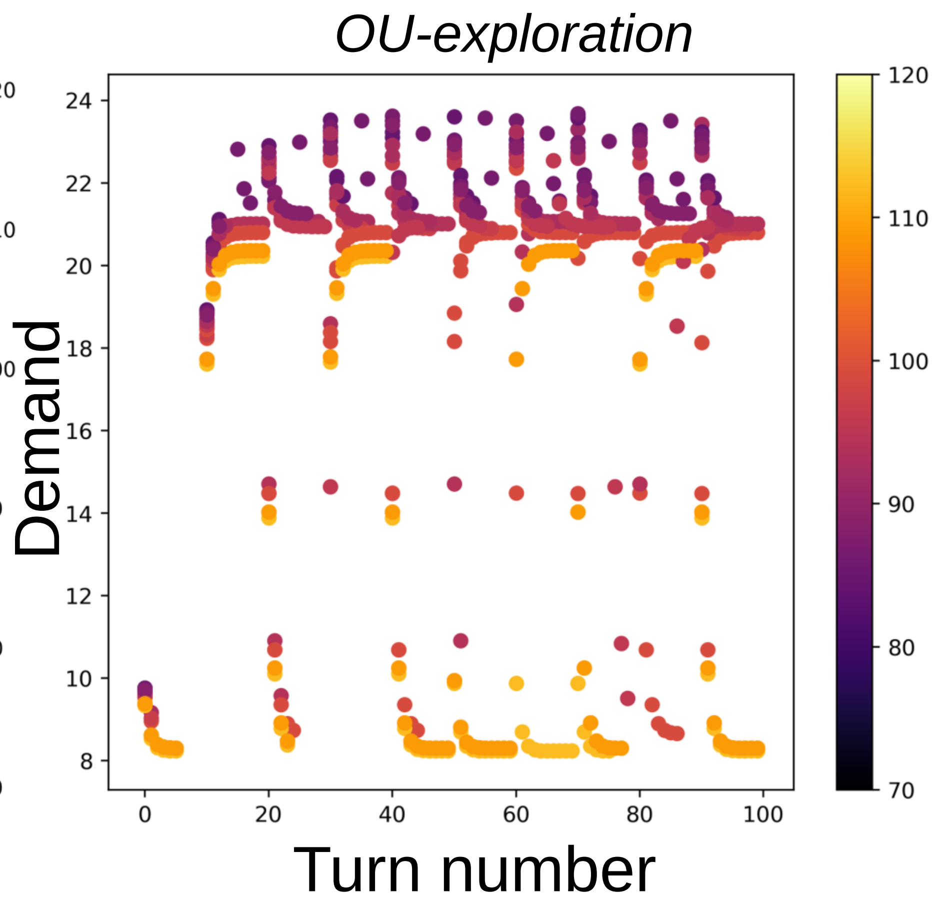
Next demand



Last unsuccessful demand

Gaussian and OU exploration policy





Earnings Performance

Agent Pool	Pool Earnings	RLA Earnings	Learning Agent
ZIA	5.23	9.27	RLA+Gaussian
ZIA	7.01	8.39	RLA+OU
ZIA	5.27	12.32	RLA+OU+anneal
LMDA	7.19	10.46	RLA+Gaussian
LMDA	7.42	10.72	RLA+OU
PAA	3.71	6.91	RLA+OU

Conclusion & Outlook

- RL agent outperformed the pool (ZIA, LMDA, PAA)
- Humans (at least their first order approximation) are much more conservative than RL
- **Future work:**
  - Include more information into observation space (currently using black box setting)
  - Training adversarial RL agents