ETH zürich

Deep Reinforcement Learning for Double Auction Processes

Batuhan Yardim

Aleksei Khudorozhkov

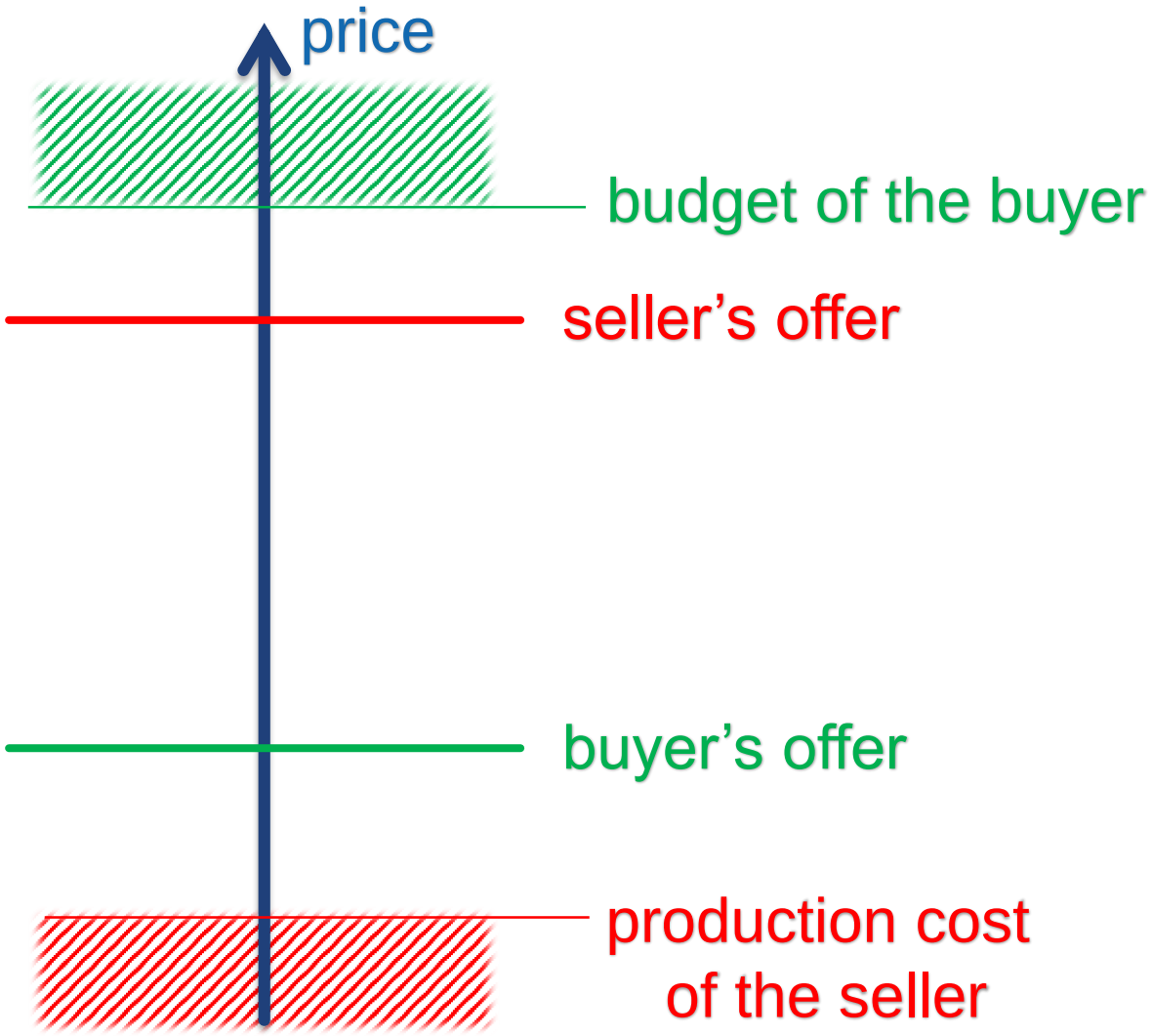
Ka Rin Sim

Neri Passaleva

Double auction

seller

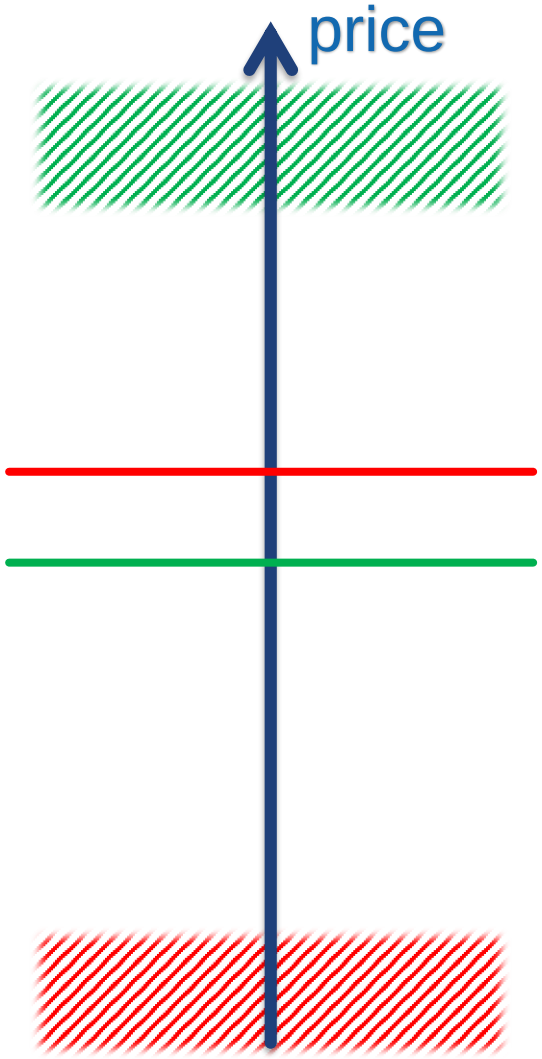
buyer



Double auction

seller

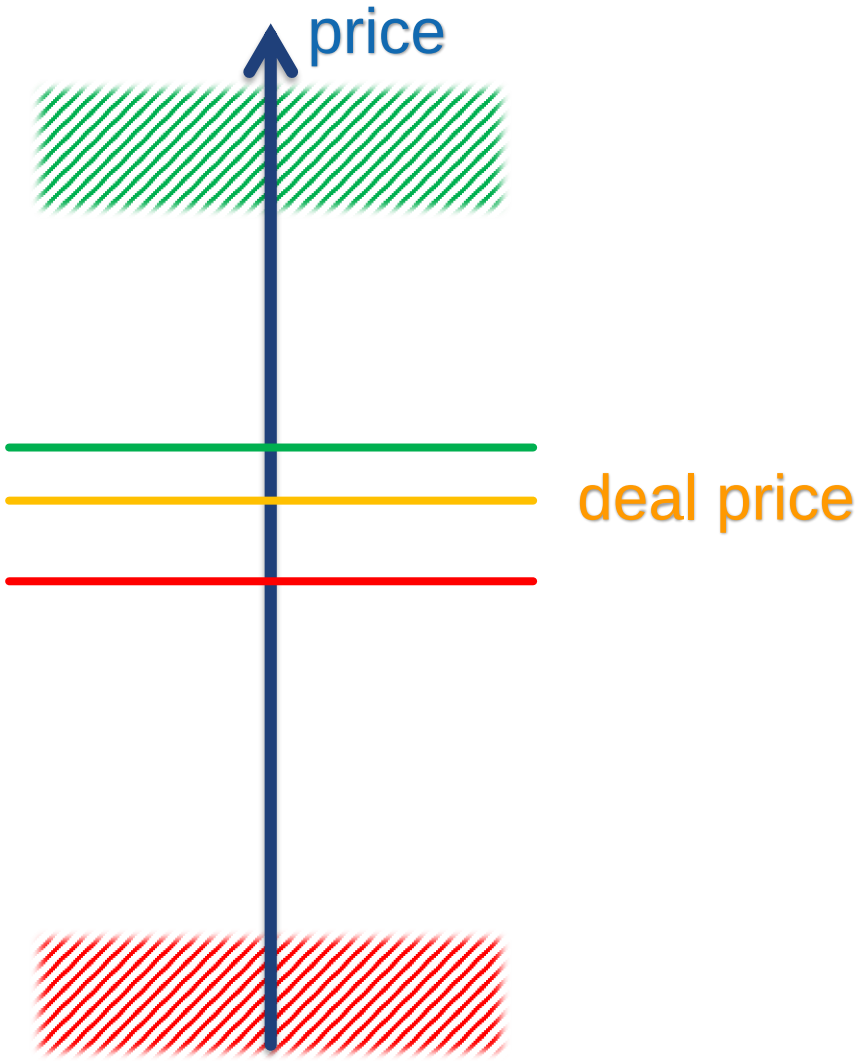
buyer



Double auction

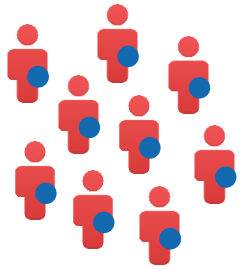
seller

buyer

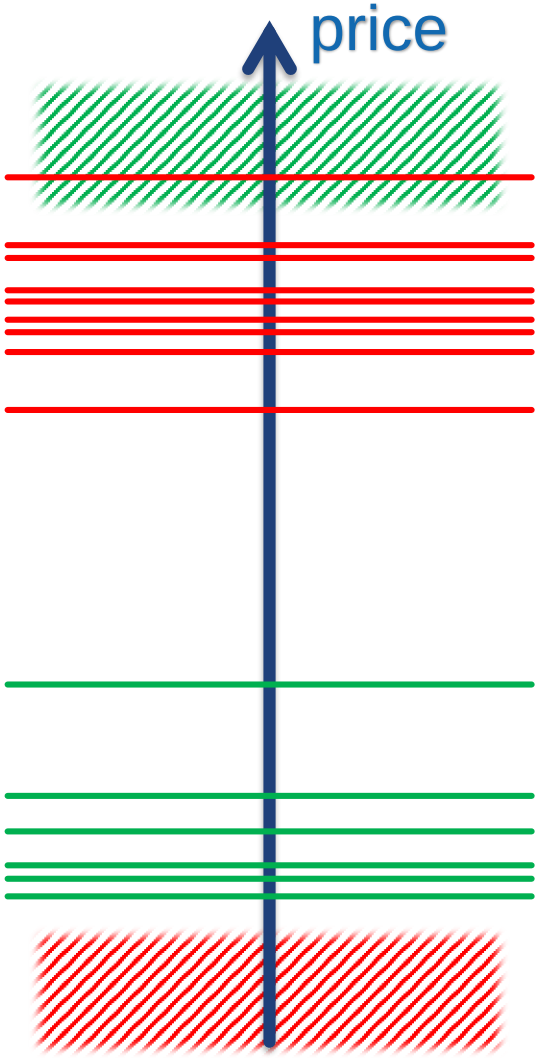


Double auction

sellers



buyers



Each agent wants to maximize the reward

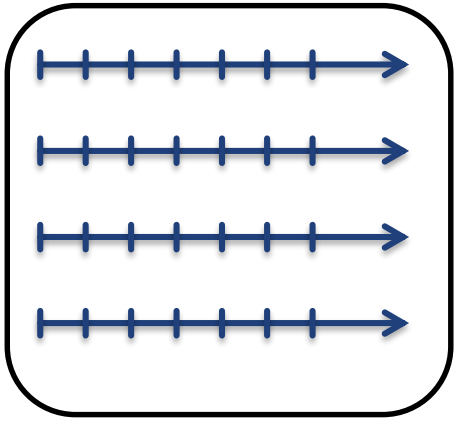
For this they can choose different strategies

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

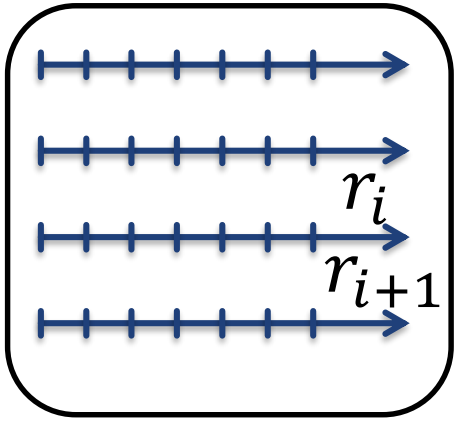
- After each time step an agent receives observations from the market environment.
- Core observations are:
  - The last offer of the agent
  - Current time step
  - bool: if the agent managed to make a deal in the previous round
- Other observations might be included:
  - Last offers of agents of the same/opposite side
  - Reservation prices of agents of the same/opposite side
  - Information about completed deals in the current round
  - The maximum time steps in a round
  - The total number of buyers/sellers
  - The number of buyers/sellers who hasn’t made a deal yet

Reward Mechanism

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

$$r_i = |p_i - a_i| \qquad \text{reward for round } i$$

- Reward is cumulative throughout the rounds



$$r = \sum_i r_i \qquad \text{total reward}$$

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 current time step

Demand at previous time step

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

Boolean indicator of previous round outcome

Noise

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

Price Aggressive Agents (PAA)

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

If **s** is **TRUE:** $$d_i = \alpha d_{i-1} + n_i$$

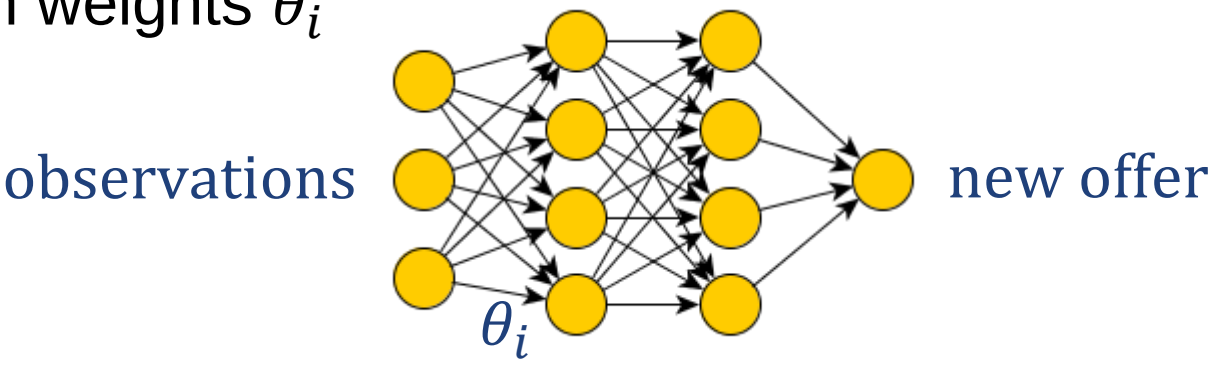
If **s** is **FALSE** $$d_i = (\alpha + \varepsilon) d_{i-1} + n_i$$

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

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

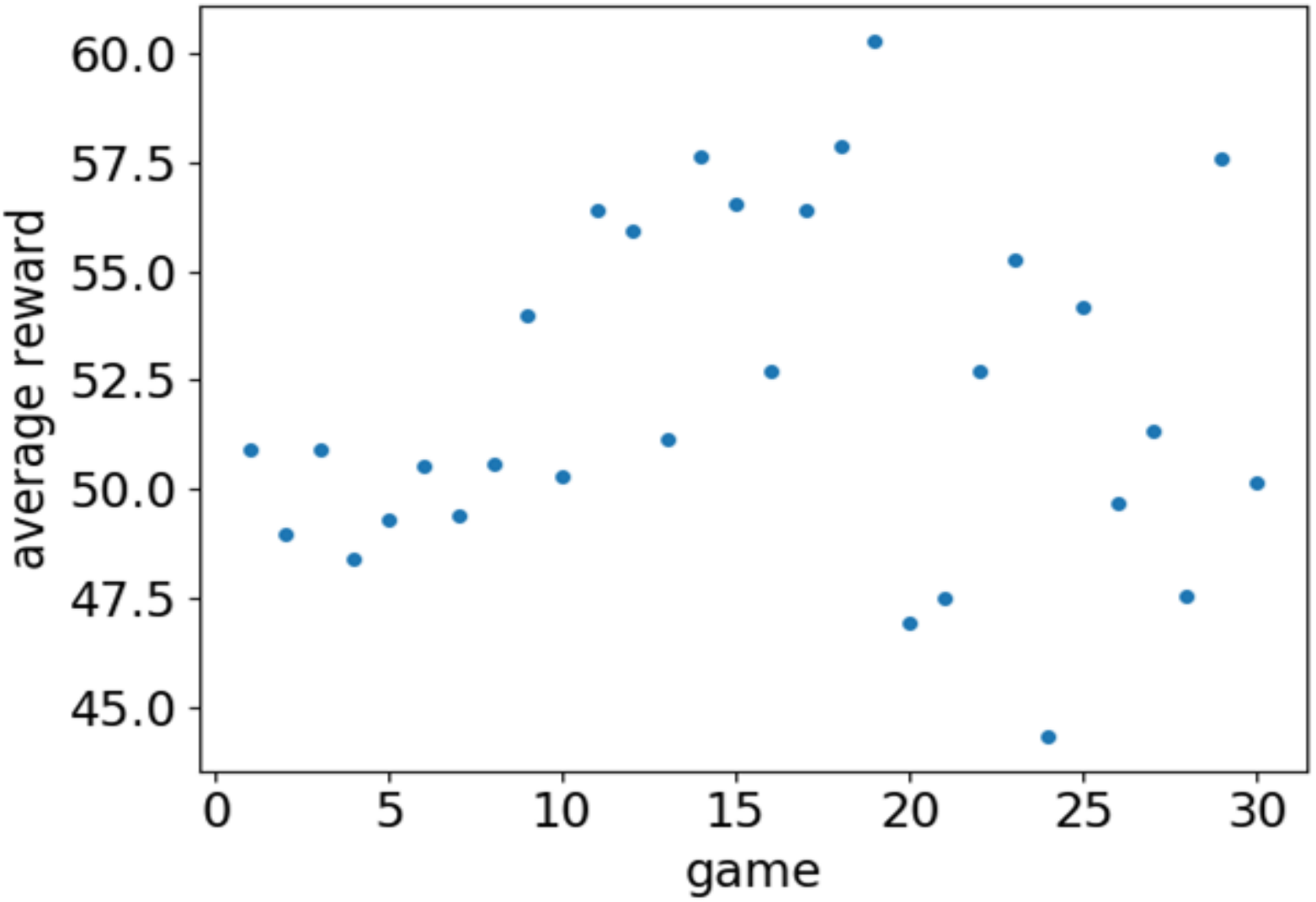
Gaussian noise

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

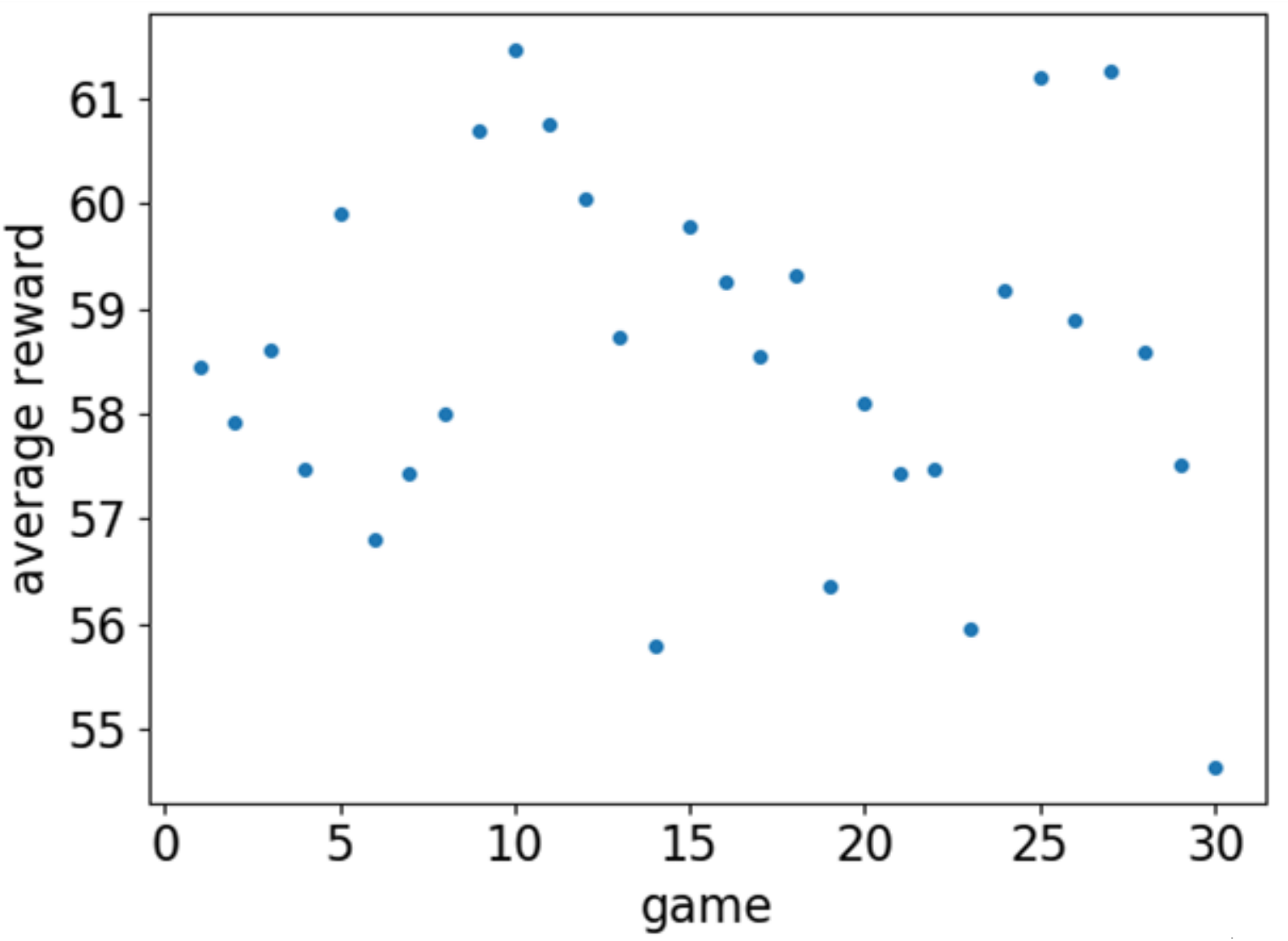
Results: Non-Intelligent Agents (ZIA, LMDA, PAA)

- No learning
- No correlation

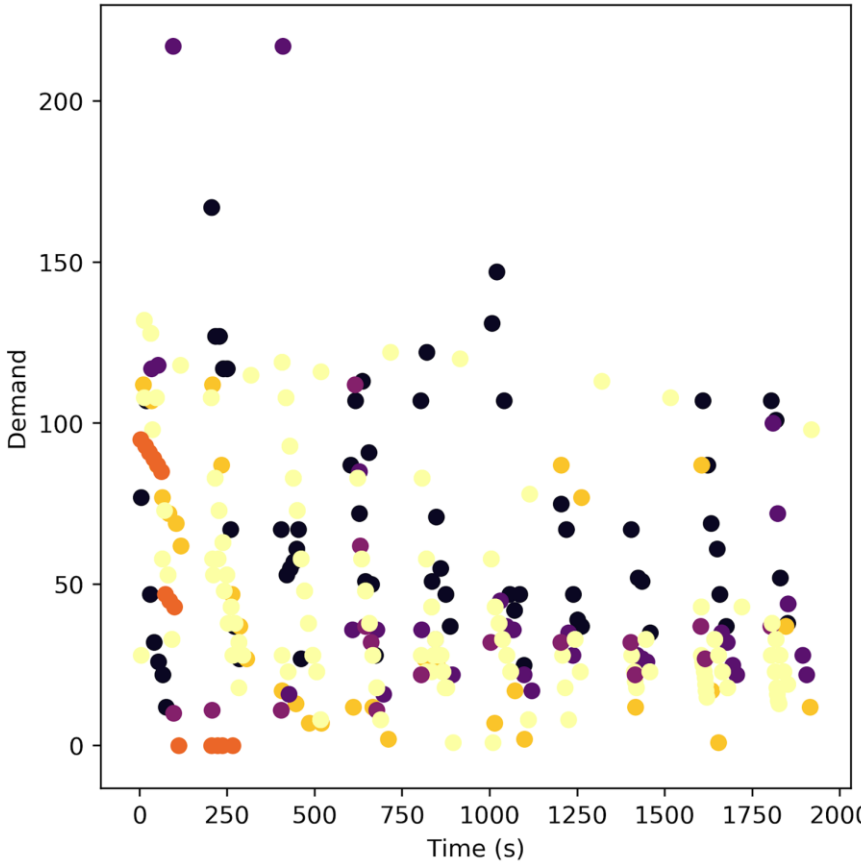
Zero Intelligence Agents (ZIA)



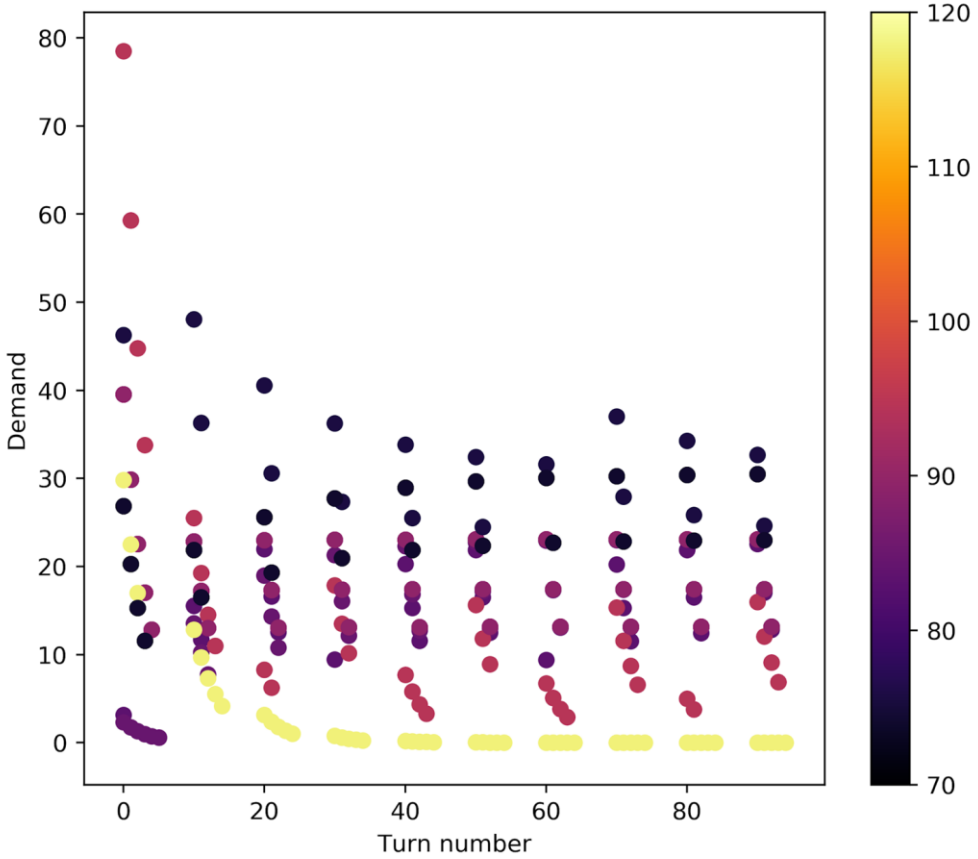
Linear Markov Decision Agents (LMDA)



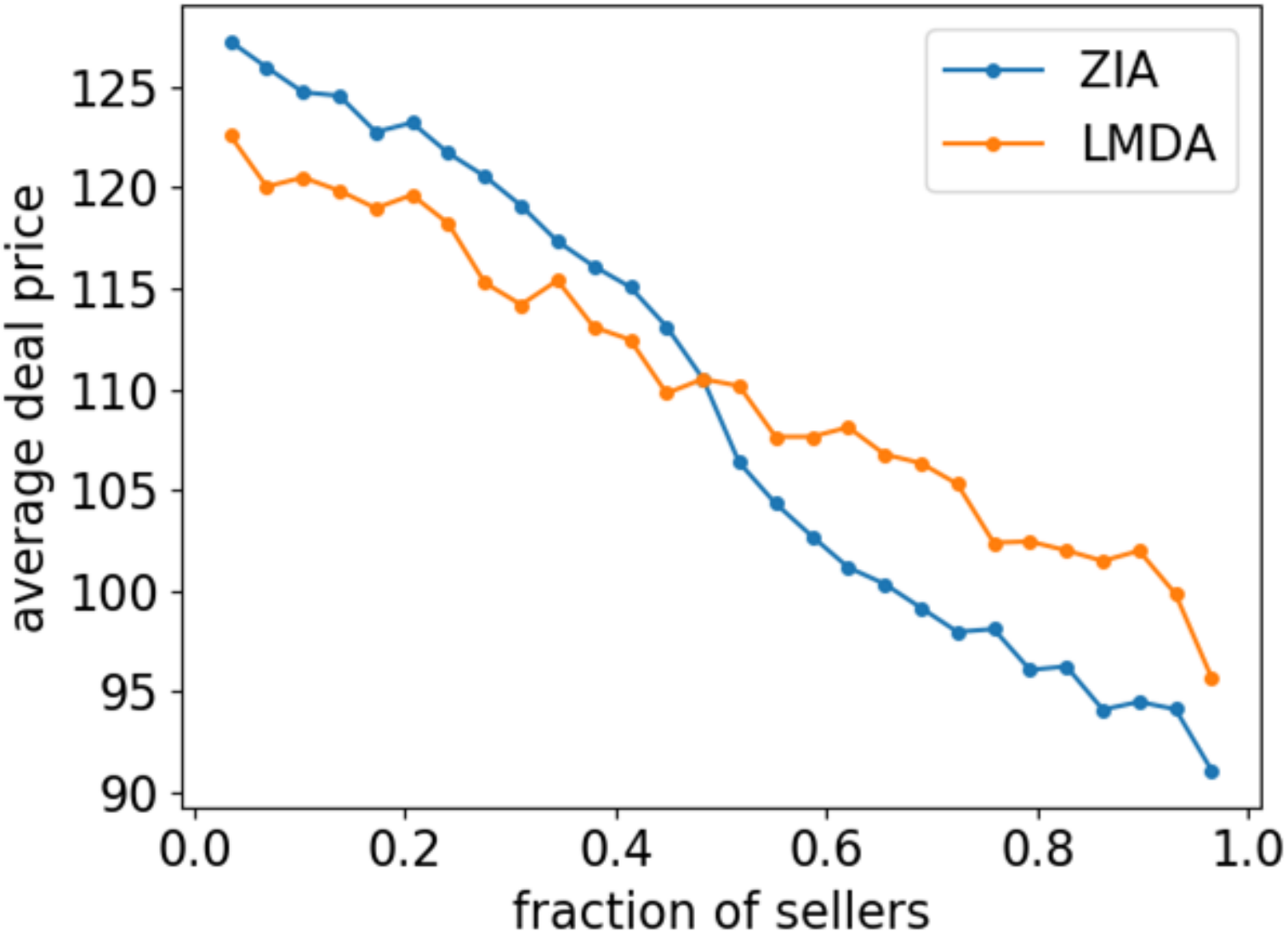
Demands -- Experiments vs LMDA



Experiments



LMDA

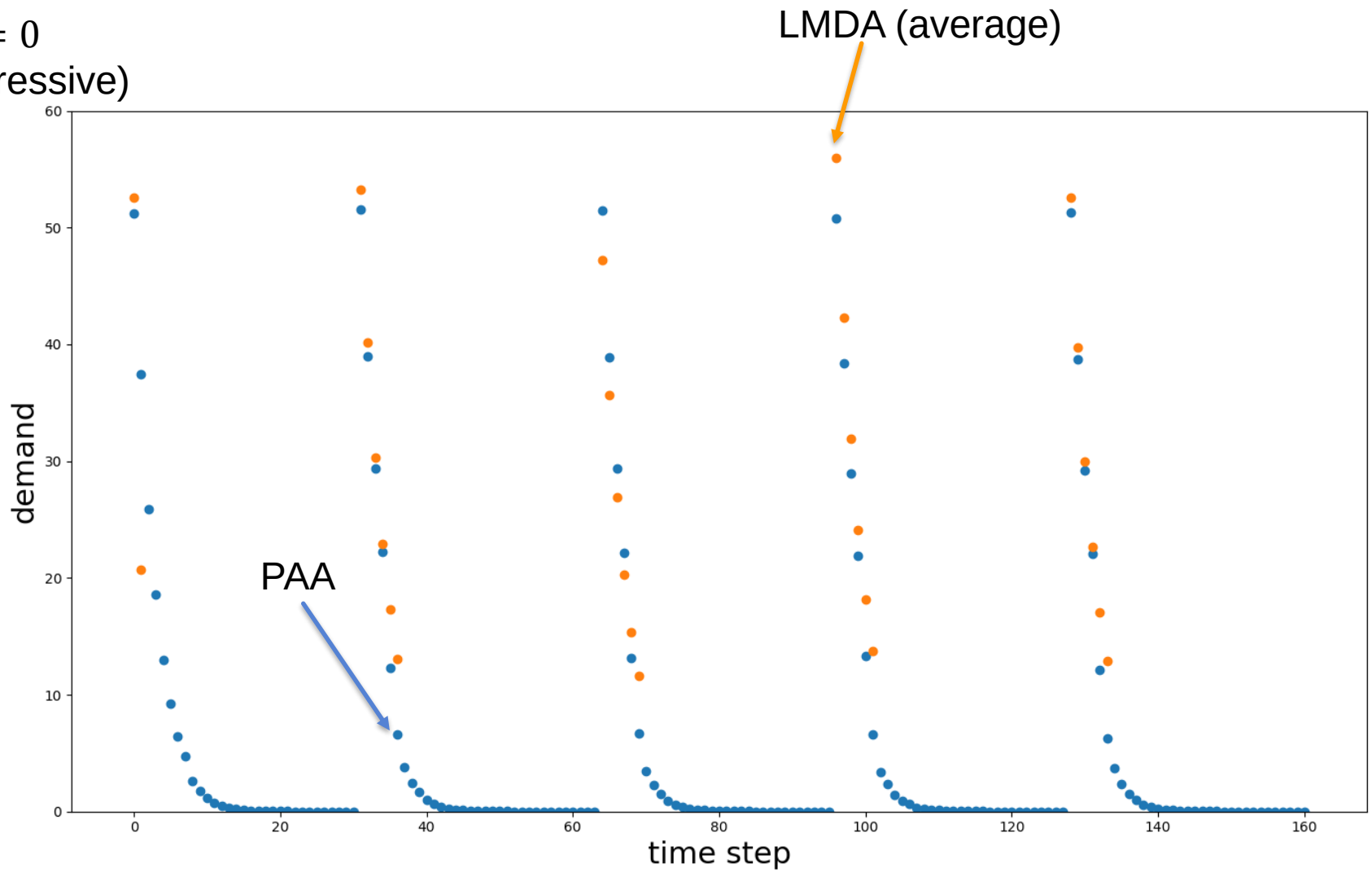

ZIA
LMDA
average deal price
125
120
115
110
105
100
95
90
0.0
0.2
0.4
0.6
0.8
1.0
fraction of sellers

Deal Price vs Fraction of Sellers

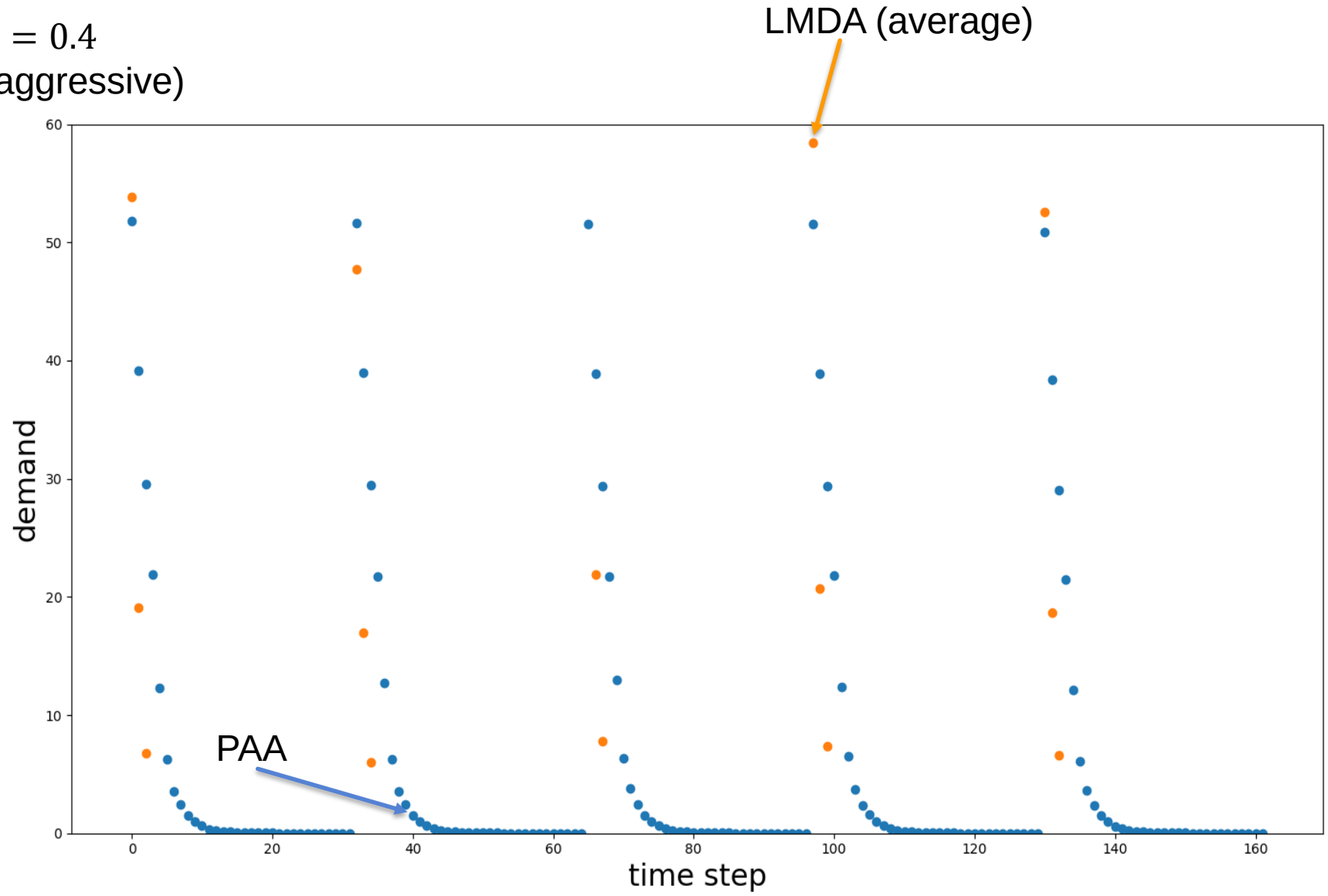
- Seller fraction increases → deal price decreases
- Competition
- Slope → competition
  - ZIA faces tougher competition than LMDA

Price Aggressive Agents (PAA)

$\varepsilon = 0$
(non-aggressive)



$\varepsilon = 0.4$
(quite aggressive)

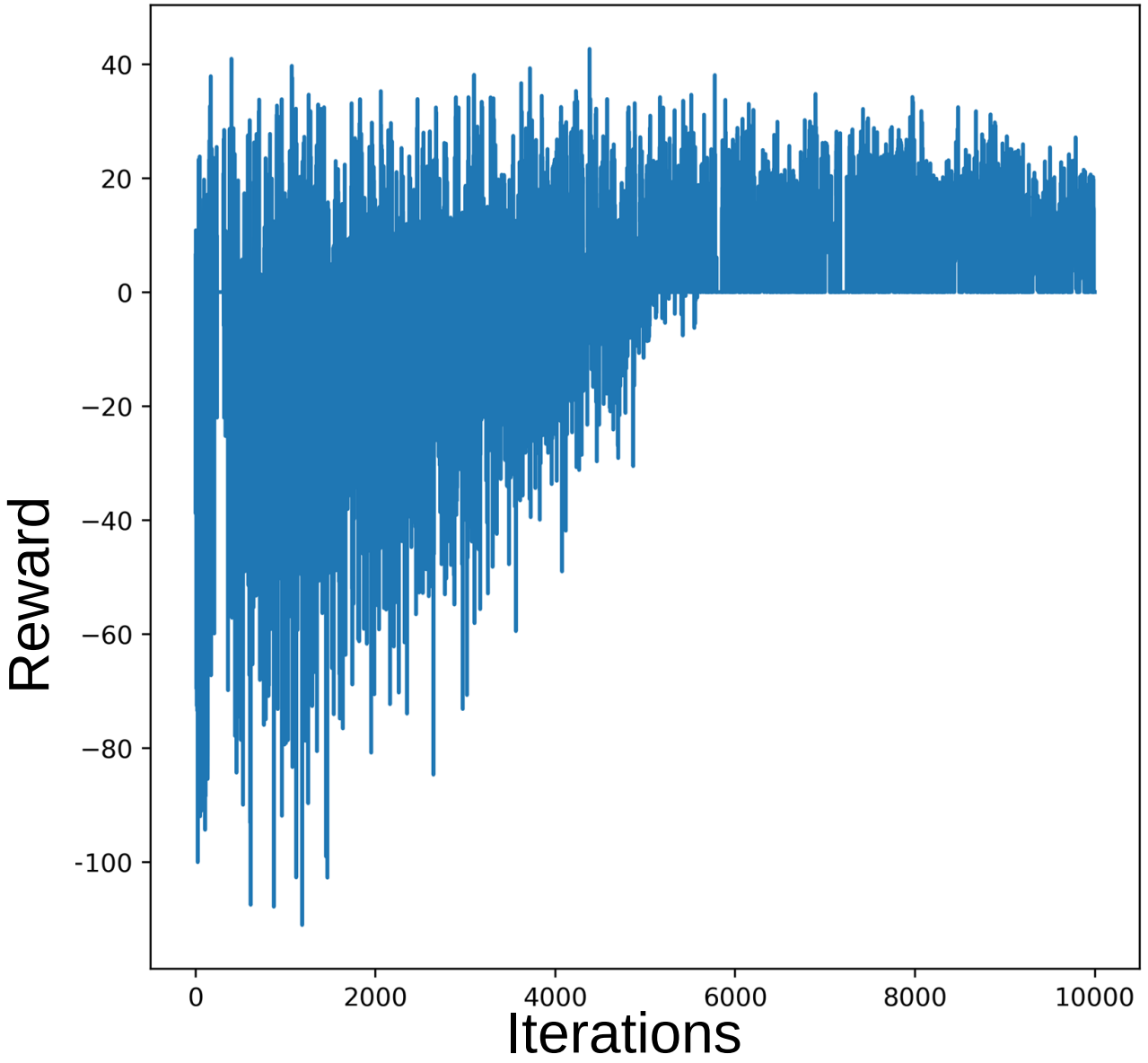


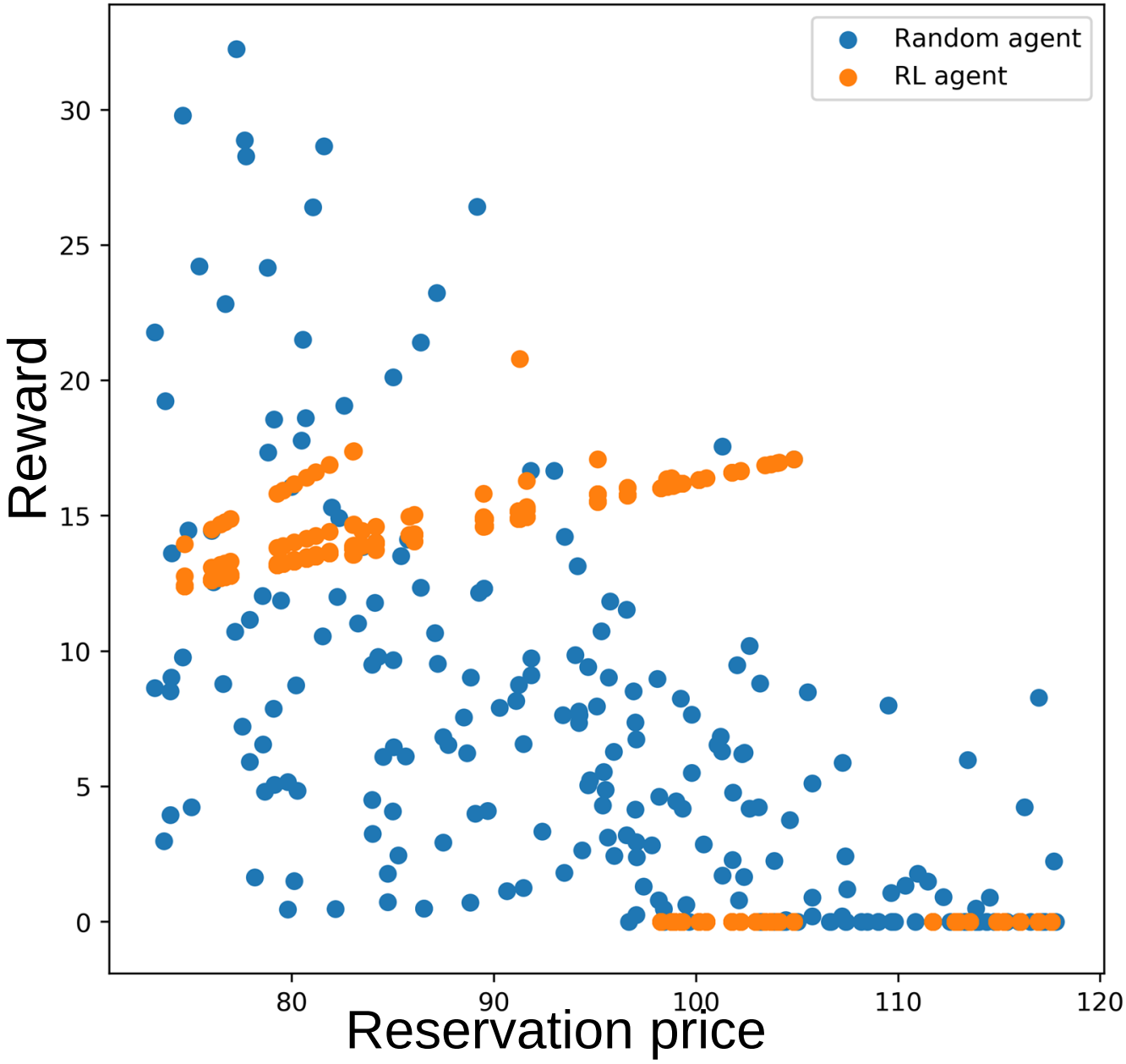
- PAA always has a lower offer price
- Deal is achieved faster by PAA

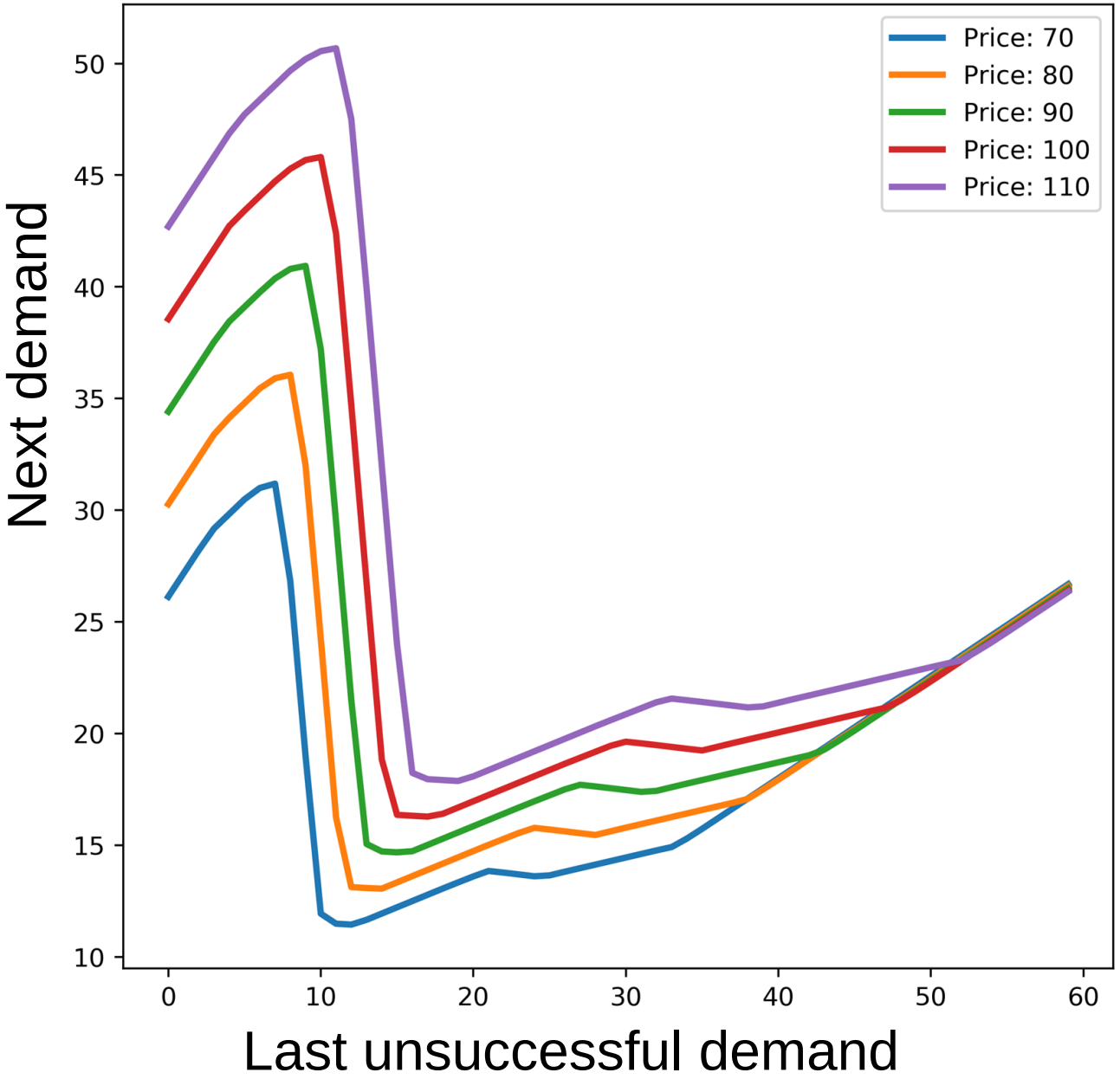
Deep Reinforcement Learning Model

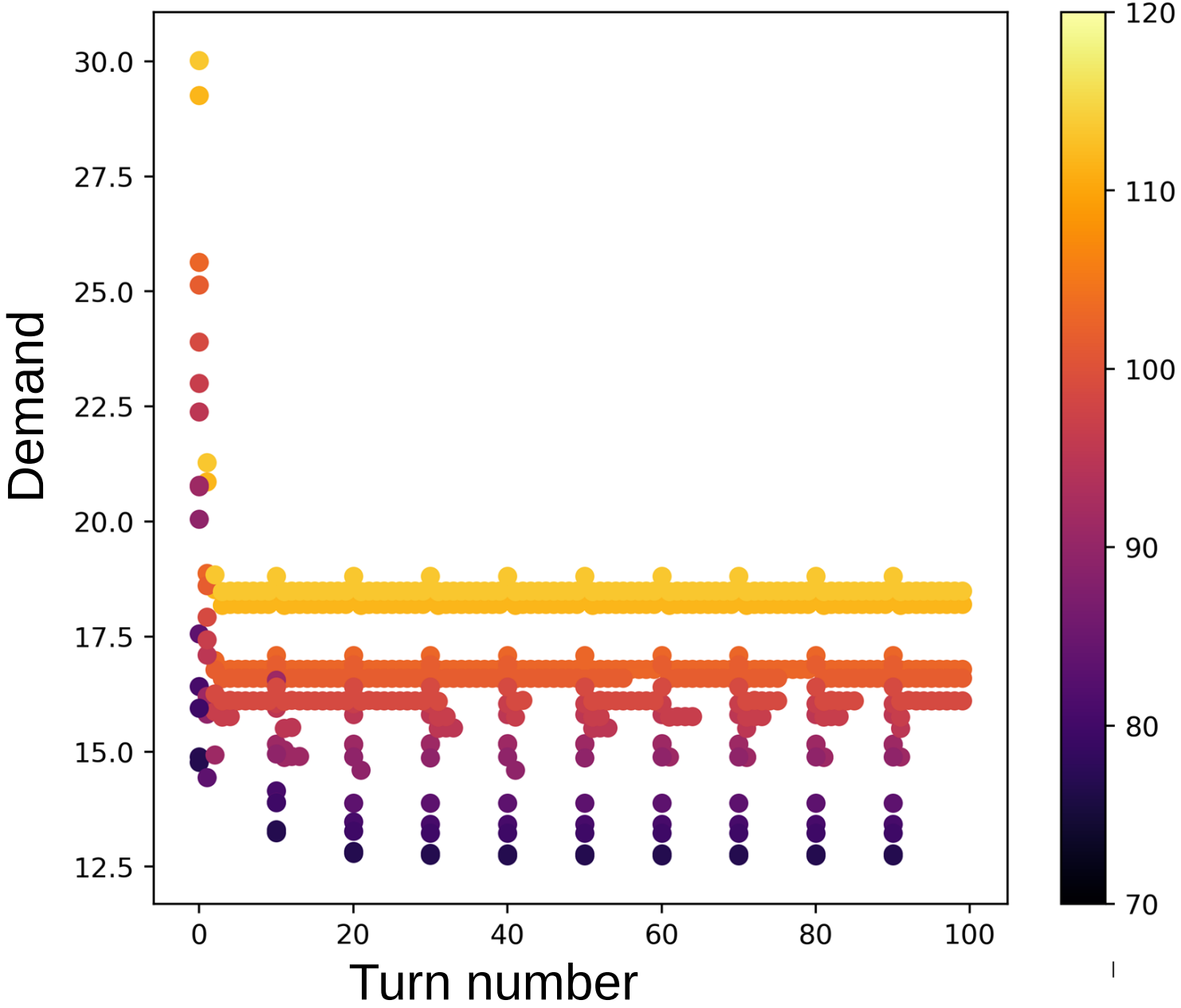
- 2 different exploration policies:
  - Gaussian
  - Ornstein-Uhlenbeck (OU)
- 1 intelligent agent + a pool of non-intelligent agents (e.g. ZIA, LMDA)

Gaussian Exploration Policy

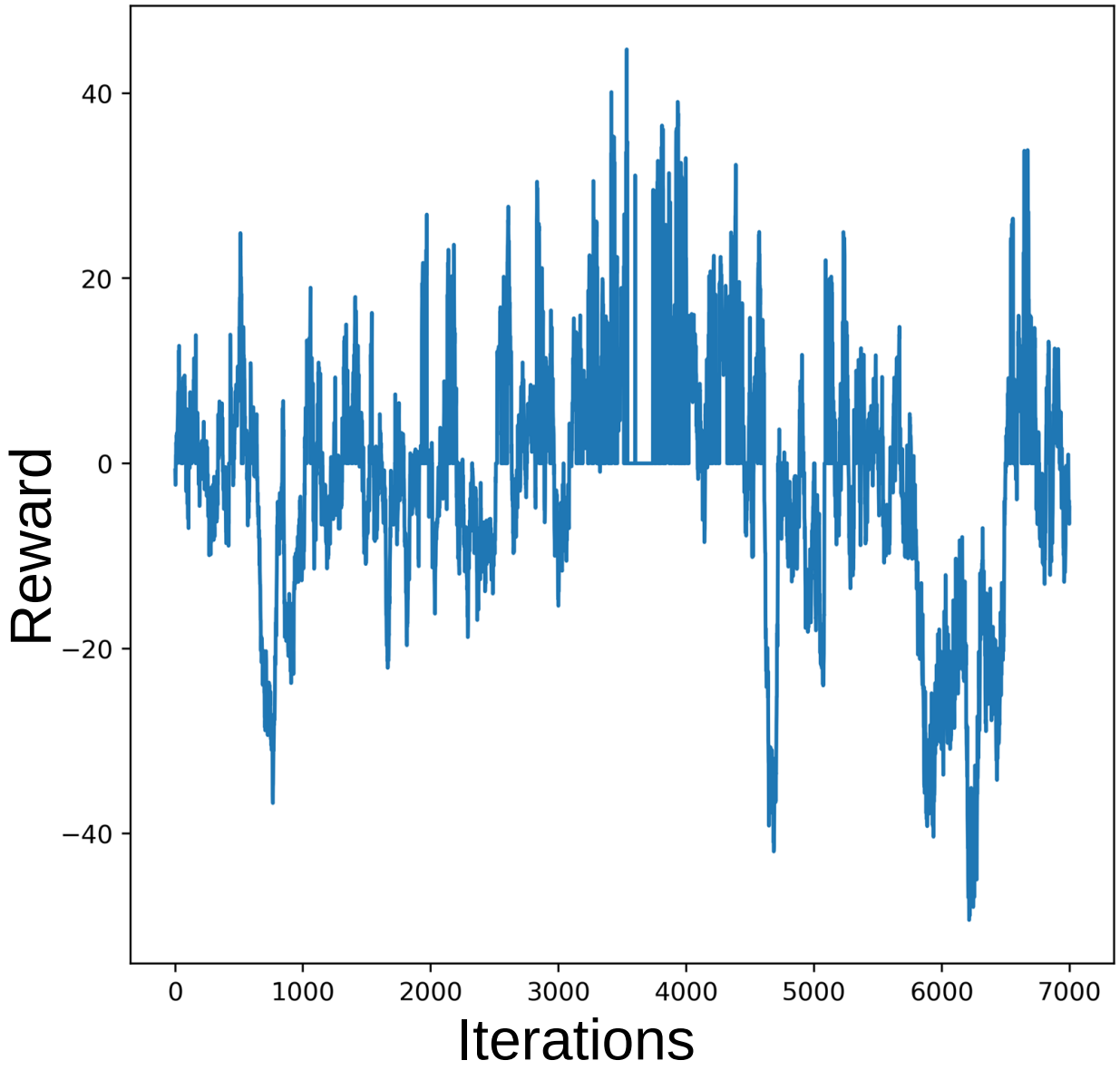

Reward
40
20
0
−20
−40
−60
−80
-100
0
2000
4000
6000
8000
10000
Iterations

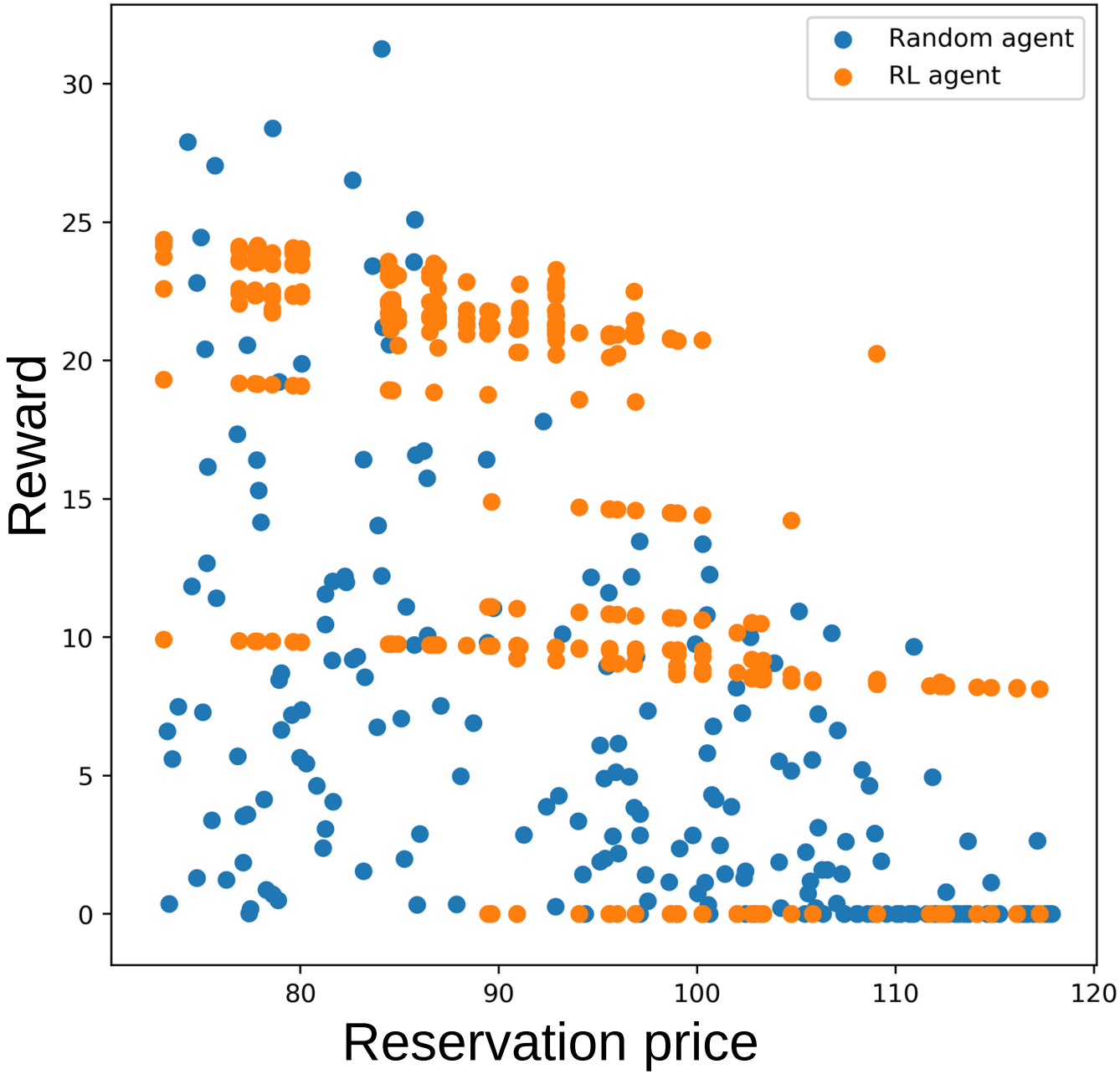

Random agent
RL agent
Reward
Reservation price
0
5
10
15
20
25
30
80
90
100
110
120

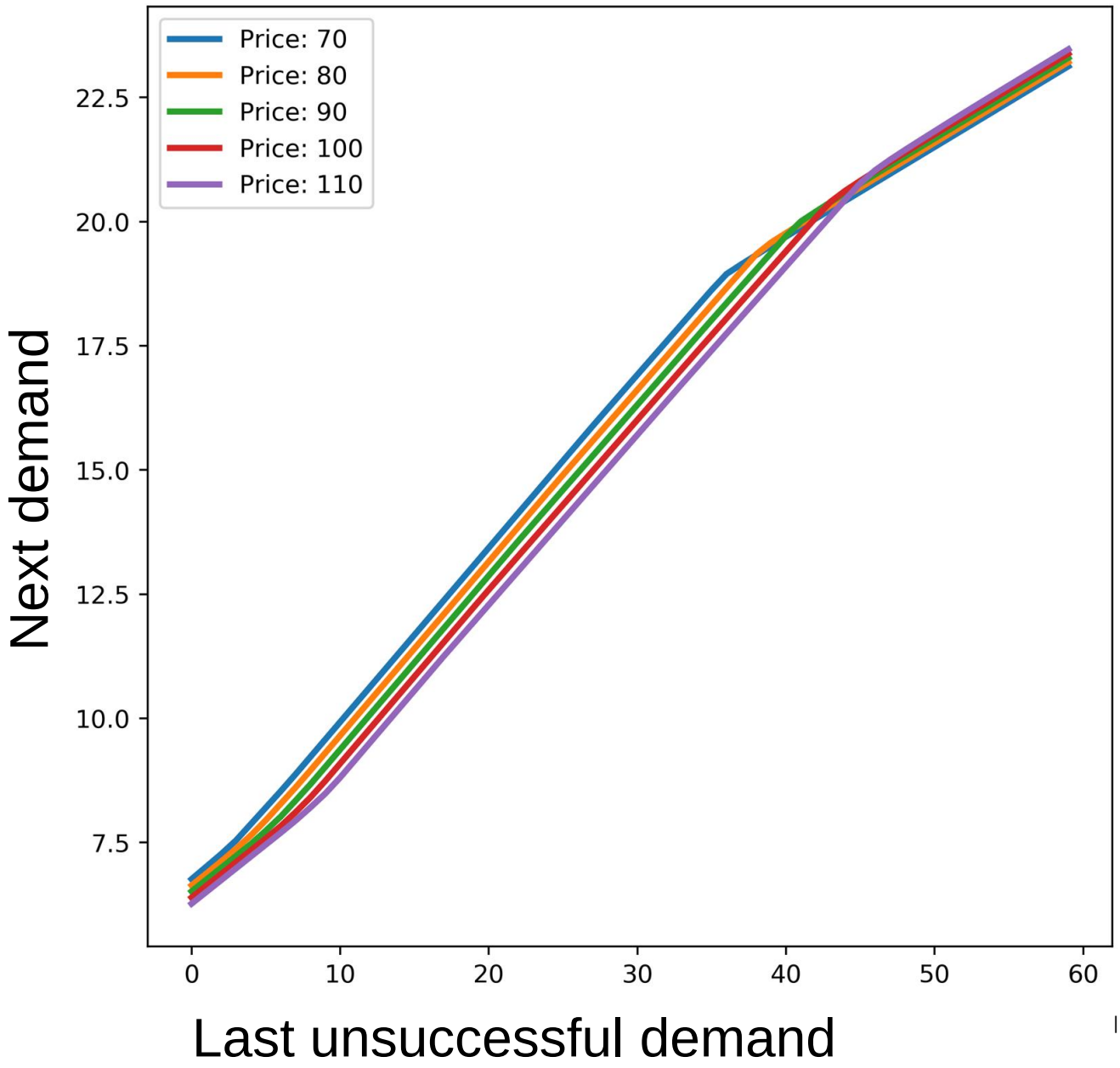

Price: 70
Price: 80
Price: 90
Price: 100
Price: 110
Next demand
Last unsuccessful demand
10
15
20
25
30
35
40
45
50
0
10
20
30
40
50
60

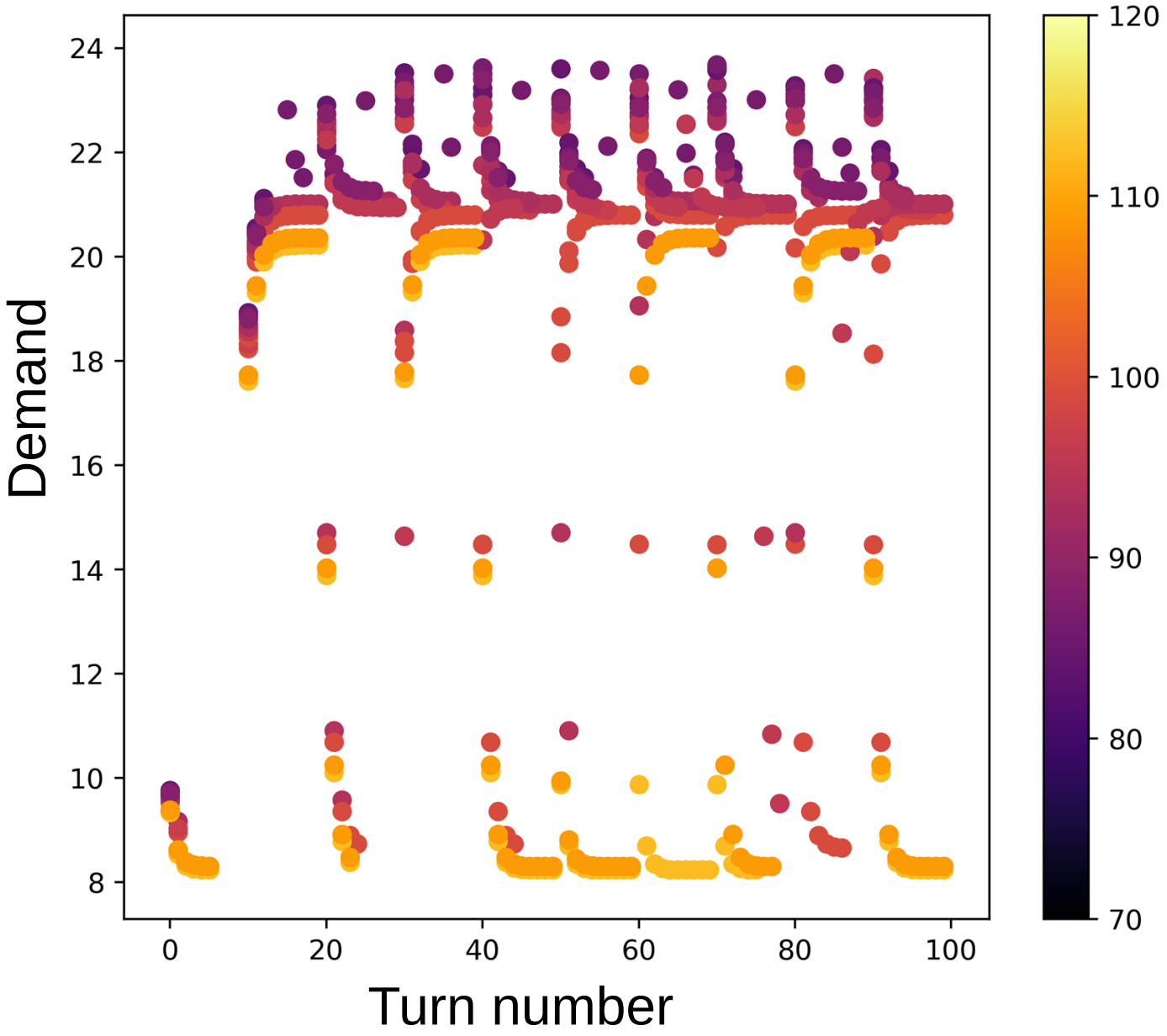

Demand
30.0
27.5
25.0
22.5
20.0
17.5
15.0
12.5
0
20
40
60
80
100
Turn number
120
110
100
90
80
70

Ornstein-Uhlenbeck (OU) Exploration Policy


Reward
40
20
0
−20
−40
0
1000
2000
3000
4000
5000
6000
7000
Iterations


Random agent
RL agent
Reward
Reservation price
0
5
10
15
20
25
30
80
90
100
110
120


Price: 70
Price: 80
Price: 90
Price: 100
Price: 110
Next demand
22.5
20.0
17.5
15.0
12.5
10.0
7.5
0
10
20
30
40
50
60
Last unsuccessful demand


Demand
24
22
20
18
16
14
12
10
8
0
20
40
60
80
100
Turn number
120
110
100
90
80
70

- reward is negative at early stages
- Gaussian agent learns quickly to avoid negative reward
- higher reservation price
  - more difficult to sell
  - market performs worse
- Gaussian agent performs better than OU agent

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	-	RLA+Gaussian
LMDA	-	-	RLA+OU
PAA	-	-	RLA+OU

Conclusion & Outlook

- RL agent outperformed the pool (ZIA, LMDA, PAA)
- RL agent (Gaussian) learned to avoid negative rewards; reward increased with iterations
- Gaussian agents have a larger profit margin than OU agents for higher res. price
- Humans are much more conservative than RL
- Extension: include more information into observation space
  - Currently using black box setting