ETH zürich

RL project from the Monte-Carlo Masters

For the course Agent-Based Modeling and Social System Simulation

We are the Monte-Carlo Masters

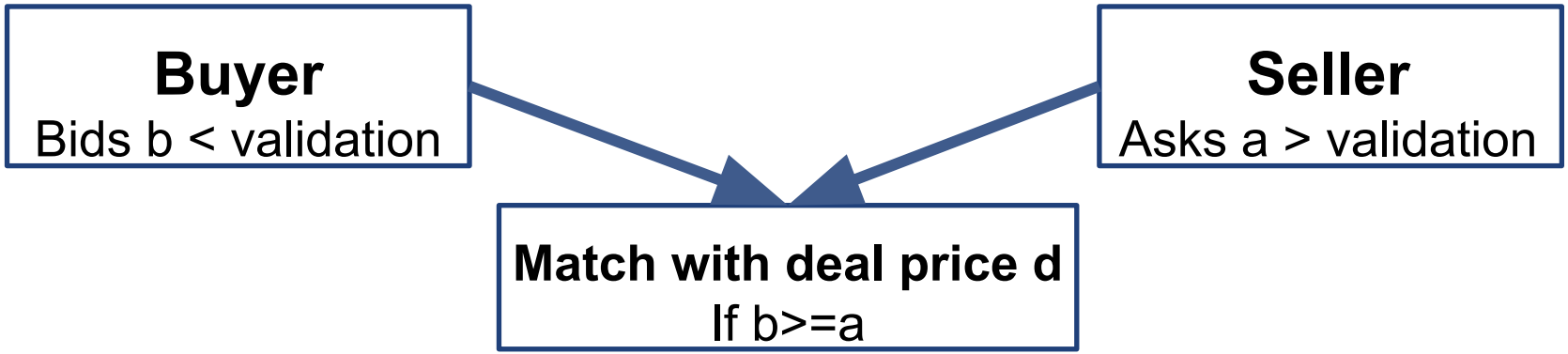
Image from: www.montecarlotennismasters.com (2019)

Unfortunately we did not play tennis together but we did a RL project.

We are:

- Shanshan
- Qifan
- Gauzelin
- Till

The double auction experiment



Information settings → a, b

- Full information setting
- Same side information setting
- Opposite side information setting
- Black-box information setting

Match Mechanisms → d

- Random matcher
- First price

Data Set

ID	side	game	round	bid	time	deal price
01	buyer	01	01	100	04	
01	buyer	01	01	115	09	115
01	buyer	01	02	105	81	
...						
01	buyer	02	01	098	06	
...						
02	buyer	01	01	107	11	
...						
10	seller	01	01	120	05	
...						
20	seller	05	10	110	962	112

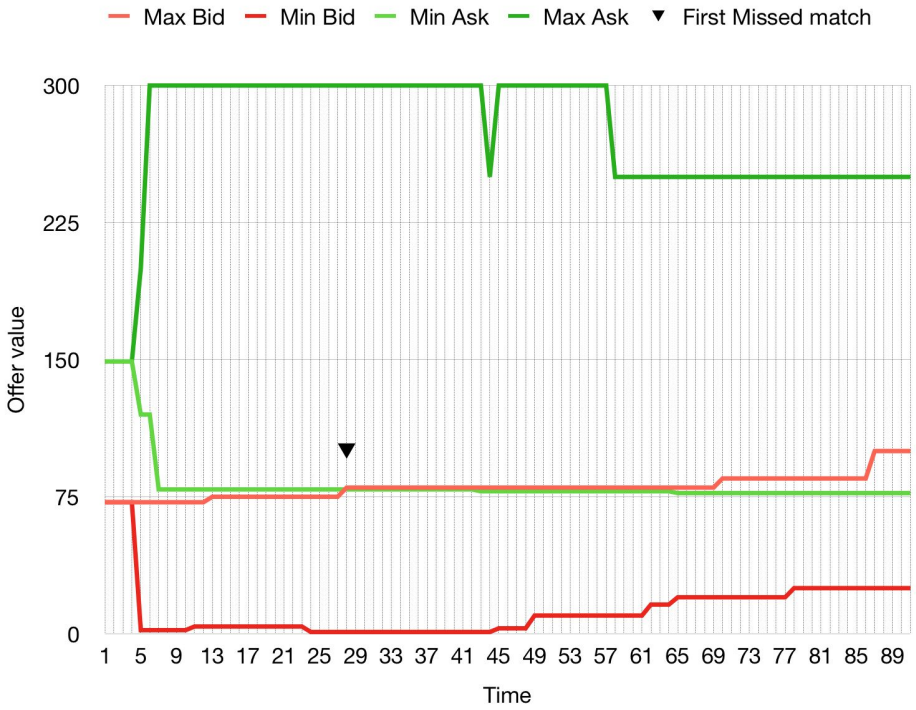
Table 3.1: mock data set

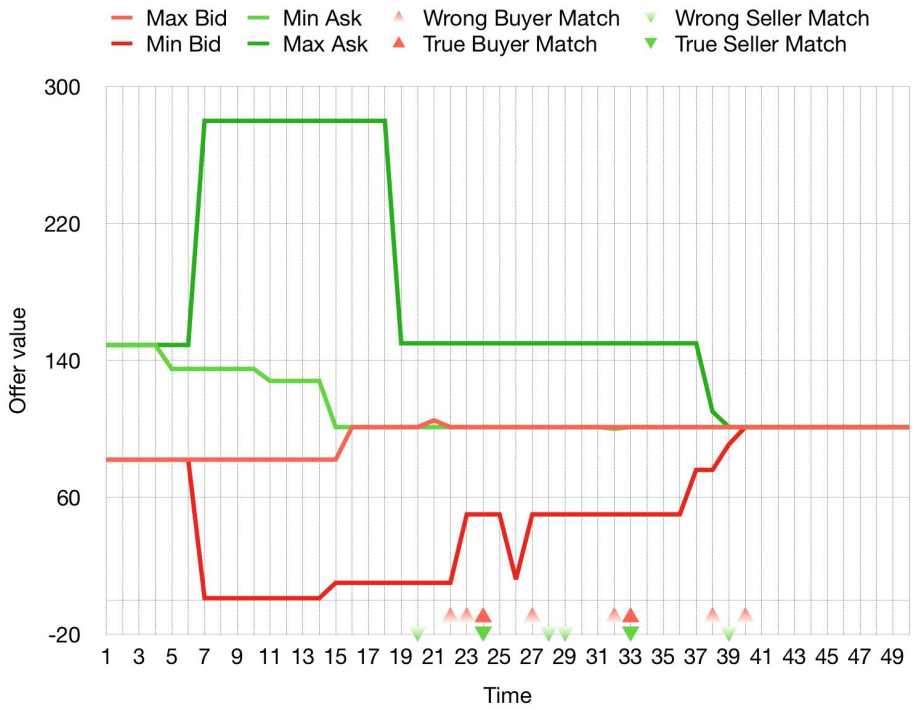
time	ID	bid
01	01	100
01	02	107
...		
01	20	130
02	01	100
02	02	107
...		
09	01	115
09	02	107
...		
90	20	115

Table 3.2: formatted data set

Data Set Discrepancies

- Bid submitted by matched agents
- Match at irrational prices
- Rational matches not happening
- One-sided matches (single agent suddenly matching)
- Outdated offers matching (over 10 seconds old offers)
- Negative deal prices
- Agents not playing at all (not technically wrong)

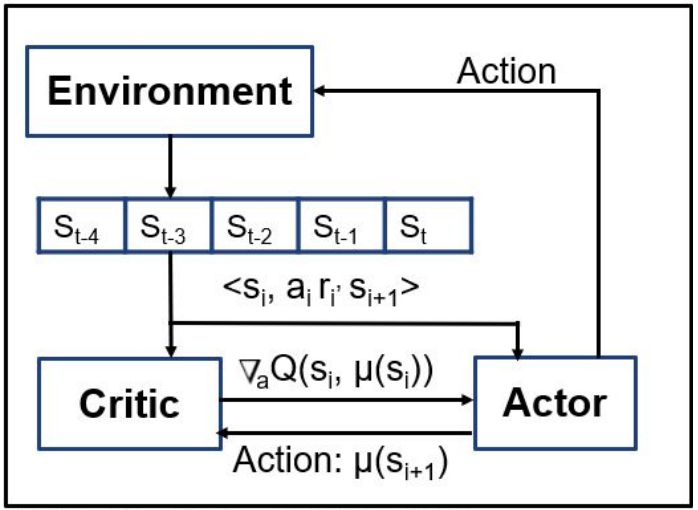

Max Bid
Min Bid
Min Ask
Max Ask
First Missed match
Offer value
300
225
150
75
0
1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89
Time


Max Bid
Min Bid
Min Ask
Max Ask
Wrong Buyer Match
True Buyer Match
Wrong Seller Match
True Seller Match
Offer value
300
220
140
60
-20
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
Time

Deep Deterministic Policy Gradients - DDPG

Overview:

- DDPG is for continuous action space
- DDPG is using a replay buffer
- DDPG is using an actor-critic approach
- DDPG is off-policy



Critic:
Input: states and actions
Output: Q(states, actions)

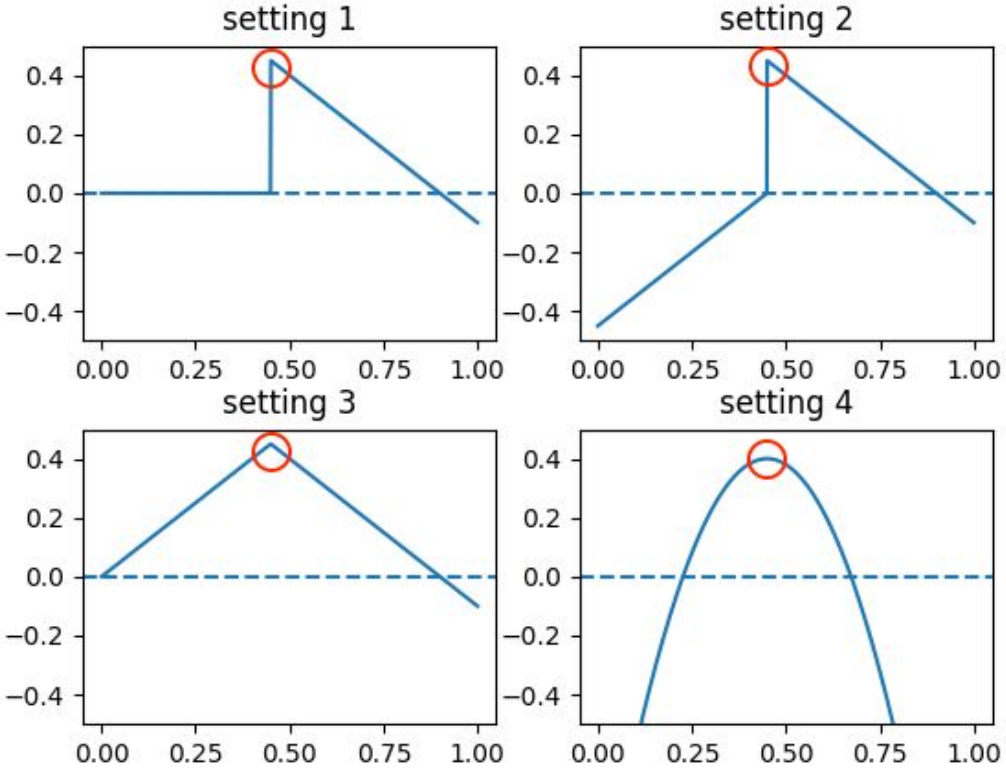
DQN for comparison:
Input: states
Outputs: Q(states, a_j) a_j: all actions

Actor:

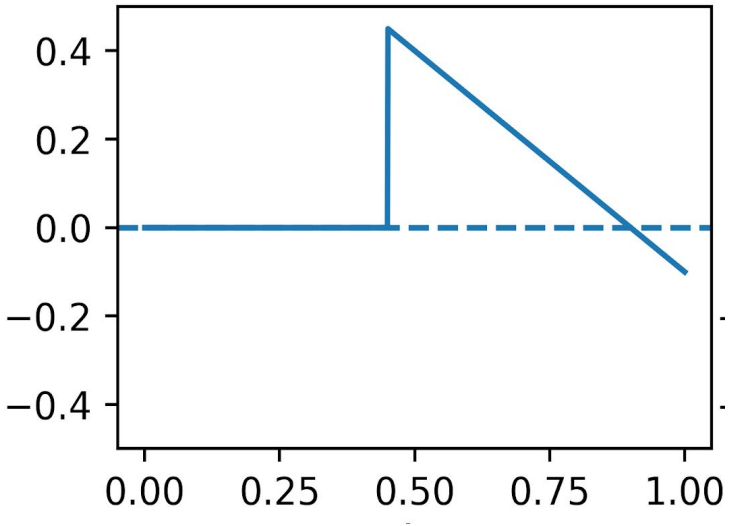
Input: states
Output: best actions

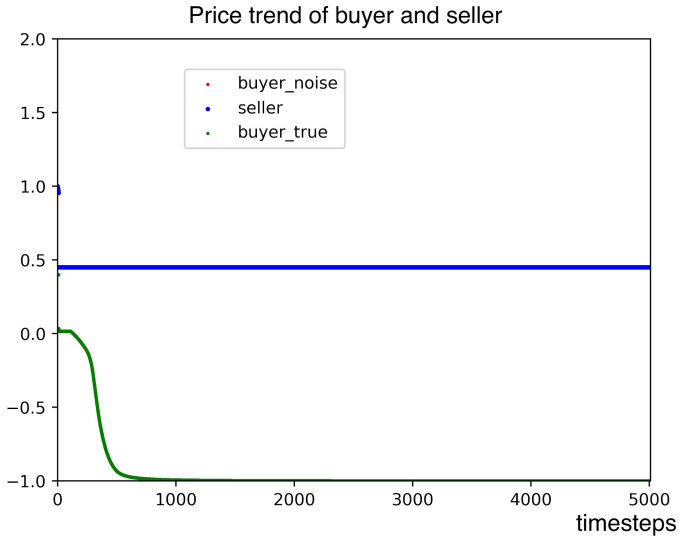
Optimizes over Q function (Critic)

Reward Function



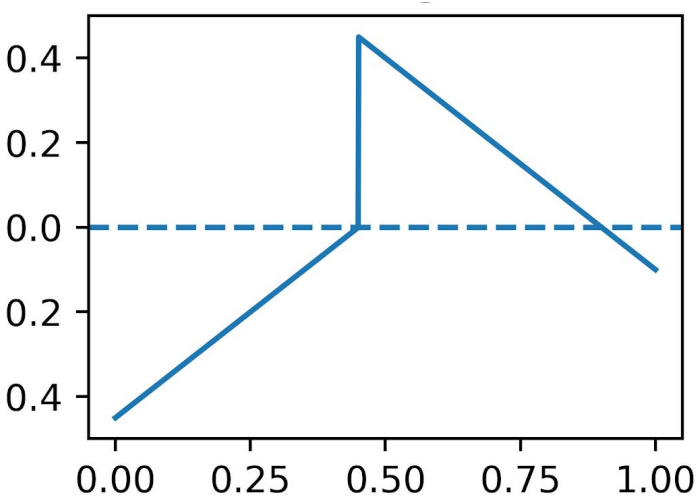
Setting 1

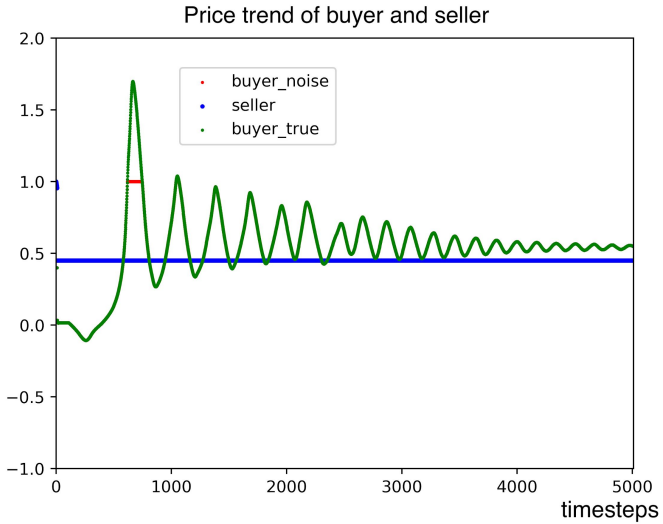




	noise=0.5	noise=0
history length 1	25.98	**579.27**
history length 10	32.72	0
history length 50	-174.42	-198.46

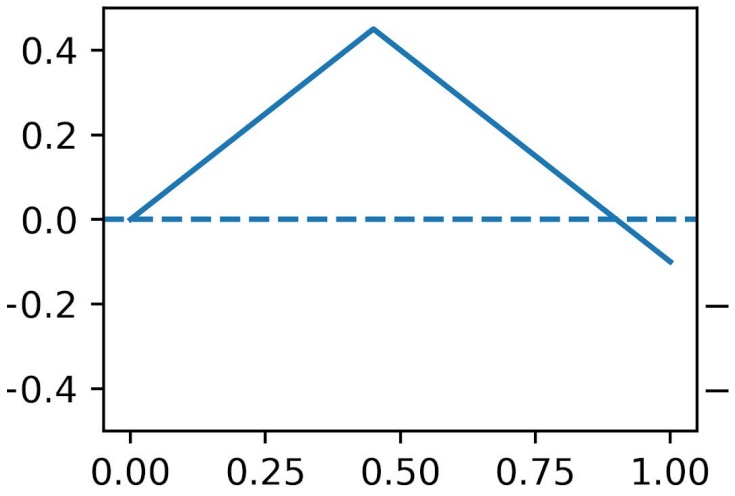
Setting 2

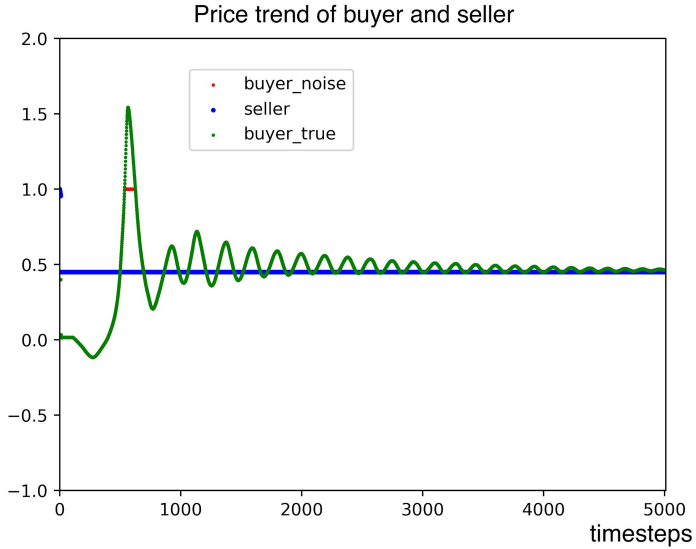




	noise=0.5	noise=0
history length 1	-235.76	-238.93
history length 10	1040.42	**1165.25**
history length 50	-160.93	-382.47

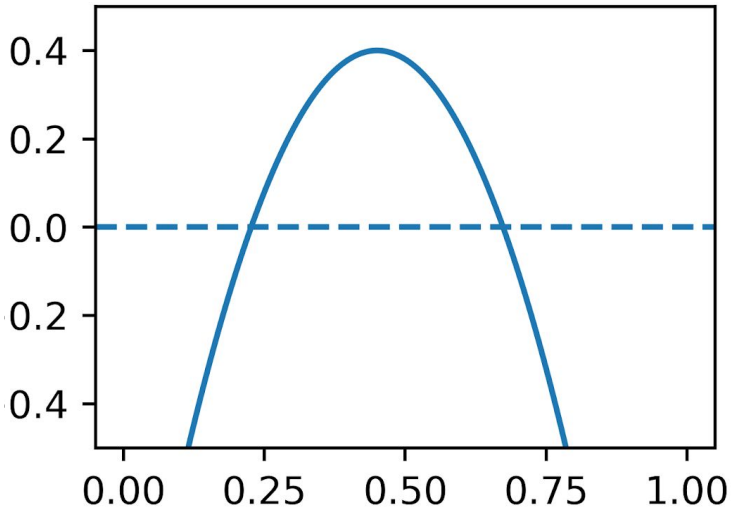
Setting 3

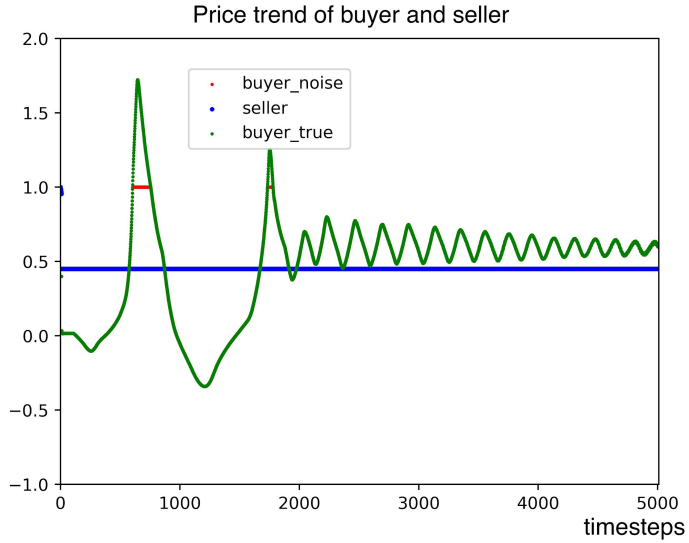




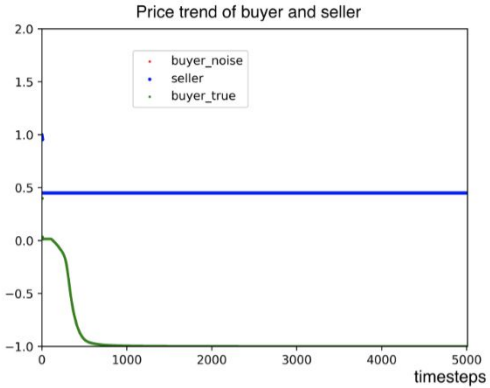
	noise=0.5	noise=0
history length 1	471.67	620.51
history length 10	-0.67	**1429.45**
history length 50	-163.13	-299.80

Setting 4

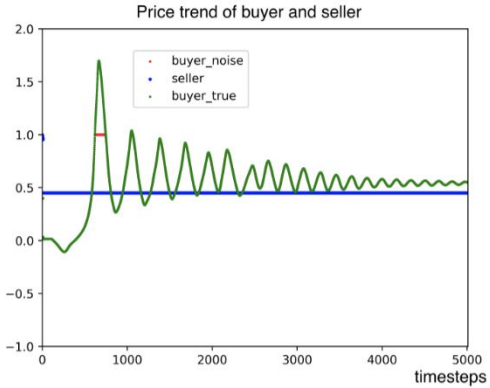




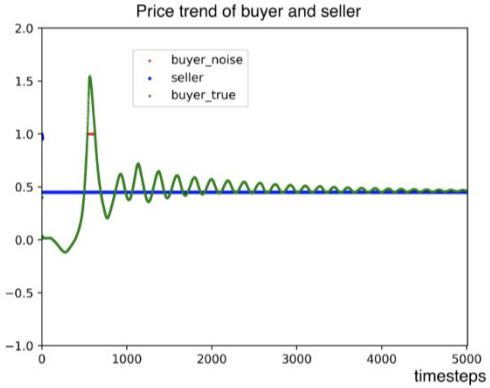
	noise=0.5	noise=0
history length 1	577.14	362.88
history length 10	215.07	**954.40**
history length 50	-308.92	-307.87



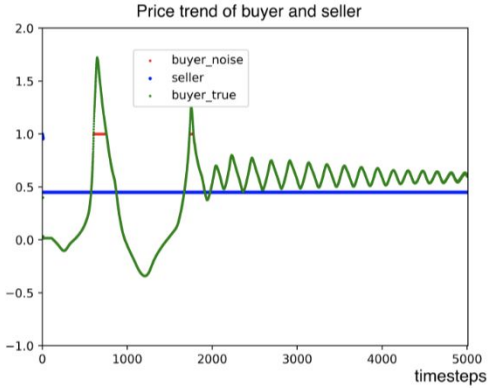
(a) setting 1



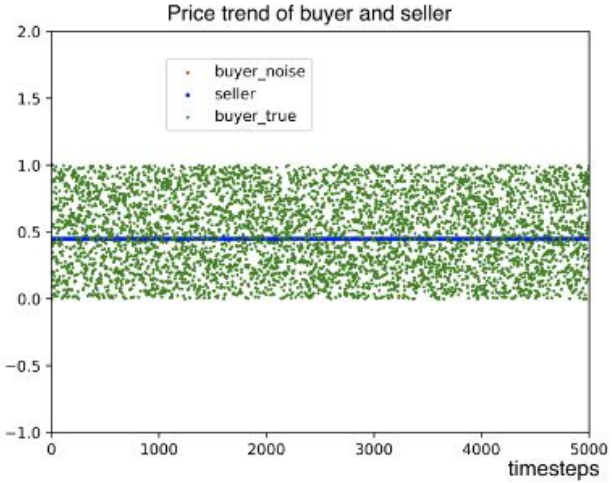
(b) setting 2



(c) setting 3



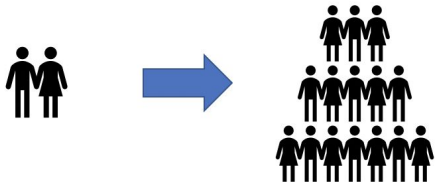
(d) setting 4



V.S. Random Agent

Future steps & Conclusions

- Successfully simulate the convergence of a single agent towards a fixed price using RL
- Needs to blend in interactions from both side of market
- Possibly extend to multi-agent case with a well defined reward function considering both sides.



Thanks for your attention

Publisher: Monte-Carlo Masters

Project folder: https://github.com/Shanshan-Huang/Monte-Carlo-Masters