Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Deep Reinforcement Learning for Double Auction Processes

Batuhan Yardim, Aleksei Khudorozhkov, Neri Passaleva & Ka Rin Sim

Zürich
6 December 2019

Contents

1 Introduction **3**

2 Models and Methodology **4**
2.1 Zero-Intelligence Agents 5
2.2 Linear Markov Decision Agents 6
2.3 Price Aggressive Agents 6
2.4 Deep RL Agents 7

3 Experiment Results **8**
3.1 Zero-Intelligence Agents 8
3.2 Linear Markov Decision Agents 9
3.3 Price Aggressive Agents 10
3.4 Deep Reinforcement Learning Model 11
3.5 Interpreting the Behaviour of the Deep Model 12

4 Conclusion **15**

5 Appendix **17**

Abstract

A double auction consists of multiple buyers and sellers submitting their offers simultaneously, with the orders executed when matching bids are paired. In this project, intelligent agents were simulated to participate in a double auction through the use of Reinforcement Learning (RL). Four types of agents were considered in this project under the black box information setting: Zero-Intelligence Agents (ZIA), Linear Markov Decision Agents (LMDA), Price Aggressive Agents (PAA) and Deep RL Agents. It was found that the agent trained using Reinforcement Learning outperformed the agent pool consisting of the non-intelligent agents, namely ZIA, LMDA, and PAA. The reward of the RL agent under the Gaussian exploration policy increased with the number of iterations, with the agent learning to avoid negative rewards. On the other hand, that of the RL agent trained under the Ornstein-Uhlenbeck (OU) process was more prone to being stuck in the negat affect their performance. It was also found that the RL agents trained under the Gaussian exploration policy has a larger profit margin for greater reservation prices, while those trained under the OU policy has reasonably concluded that making smaller profits for higher reservation prices is better. In conclusion, it was found that the RLA demands higher profit margins more confidently compared to humans especially when the reservation price is low. A possible extension to this work would be the inclusion of the additional information into the observation space of the intelligent agent.

1 Introduction

A double auction consists of multiple buyers and sellers submitting their offers simultaneously, with the orders executed when matching bids are paired (1). This is widely used in financial markets, such as the New York Stock Exchange (2), and more recently, in cryptocurrency trades. The goal of this project is to implement intelligent agents to participate in a double auction through the use of Reinforcement Learning (RL). Four types of agents were considered in this project: Zero-Intelligence Agents (ZIA), Linear Markov Decision Agents (LMDA), Price Aggressive Agents (PAA) and Deep RL Agents.

The employment of Reinforcement Learning (RL), a reward-based learning in which the algorithm maps the observations to actions in order to maximize the rewards (3), in such trades offers a promising prospect in optimizing the gains from the trades (4). The advantage of RL compared to supervised and unsupervised learning is overcoming the limitations of financial data, as the agents are free to learn the best possible strategy through exploitation of the current knowledge (state) and the exploration of of the given action space (5).

The interest in using Reinforcement Learning (RL) for simulations of financial and stock markets has been growing in the recent years (6), (7). In particular, neural networks were trained as RL agents to simulate the foreign exchange market (8). Under this framework, it

was shown that a stable learning which is generalisable to unseen data was obtained.

2 Models and Methodology

The market environment considered in this project consists of agents, which are either sellers or buyers, each with a fixed reservation price corresponding to the production cost and the budget respectively (9). The reservation prices of the buyers, b_i were sampled randomly from a uniform distribution with the range $103 \leq b_i \leq 148$ and those of the sellers, c_i from a uniform distribution with the range $73 \leq c_i \leq 118$, in accordance with the empirical data obtained in (10). The reservation price of each agent remains constant throughout a game.

An offer (or an action at time step i, which we denote by a_i) is submitted by each agent at discrete time steps, and a match is made between a buyer and a seller when the buyer's offer is higher than or equal to the seller's. The deal price of a match is taken from a uniform random distribution ranging between the seller's and the buyer's offer prices. The resulting reward r_i is then found to be the difference between the agent's offer price, a_i and their reservation price, p_i, such that

$$r_i = p_i - a_i \tag{1}$$

for buyers and

$$r_i = a_i - p_i \tag{2}$$

for sellers.

A round is terminated either when the maximum time step is achieved, or when no further matches are possible. The market environment is reset after each round. However, the reward is cumulative and is reset to 0 only after the whole game is completed.

The core information included in the observation space of every agent is their own offer from the previous time step, a Boolean representing whether they made a deal in the previous round, and the current time step. Additional information could be chosen to be added to the observation spaces, such as: the last offers of the other agents of each role, the reservation prices of the other agents of each role, the prices of completed deals in the current round, the maximum time steps in a round, the number of agents of each role and the number of agents of each role who hasn't made a deal yet in the current round.

For the sake of convenience, we define the notion of demand of an agent at time i (denoted d_i) as how much of a profit it asks as a buyer or a seller. In other words, at each time step, an offer a_i was generated such that

$$a_i = p_i - d_i \tag{3}$$

for buyers and

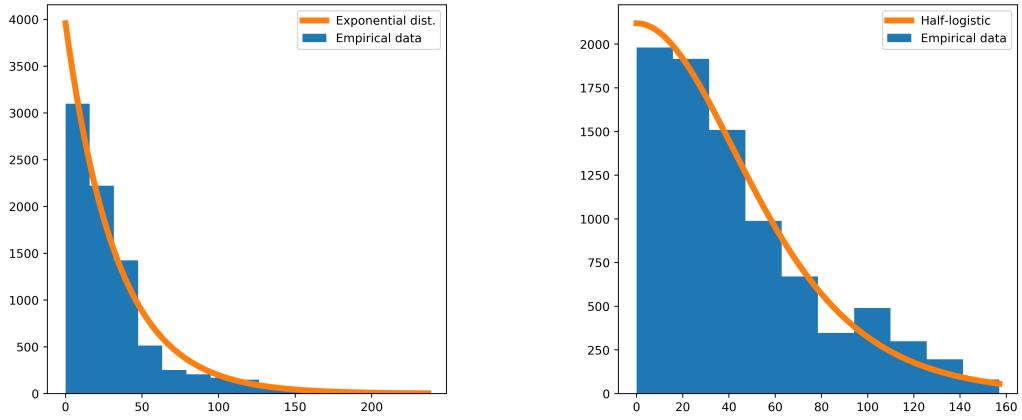
$$a_i = p_i + d_i \tag{4}$$

for sellers, where p_i is the reservation price and d_i is the demand. We relax the problem slightly, and allow agents to make negative demands $d_i < 0$, possibly receiving negative rewards. This simplifies our models and serves the purpose of numerical stability when performing reinforcement learning. However, a proper agent model will always have positive demand and positive offers. We use this fact as a sanity check in our proposed models and learning algorithms.

2.1 Zero-Intelligence Agents

The first and most simple agent class, dubbed "zero intelligence agent" (ZIA), operates under a complete black box setting, under which the core information in the agents' observation spaces is not used in the generation of new offers.

In other words, at each time step t, the ZIA sets its demand independently and identically distributed with a probability density function $\mathcal{P}$. For the purpose of modelling, a probability distribution function was fit to the experimental data of initial bids (at the first time step of the first round) extracted from human players (10). This in fact models how human agents bid when they have no observations of the environment and have not received any implicit feedback via rejected or accepted offers yet.



(a) Seller demands

(b) Buyer demands

Figure 1: Histogram of experimental buyer and seller demands in human experiments. Also superimposed is our model distributions.

By analyzing the experimental data (10), the demand of a ZIA buyer was sampled from the half-logistic distribution and a seller ZIA with the exponential distribution, both with support set $[0, \infty)$. These choices were made partly by qualitatively analysing the demand histograms and fitting the parameters of a family of distributions using numerical optimization. We also justify our choice of distribution by picking the distributions using the Kolmogorov-Smirnov test for goodness of fit.

2.2 Linear Markov Decision Agents

A slightly more complicated model, dubbed "Linear Markov Decision Agents" (LMDA), has a memory of its last state (justifying the name Markov) and can linearly regress its next demand using its observations and state. Namely, following the empirical methodology of (10), the demand at time i is specified by

$$d_i = \alpha_1 d_{i-1} + \alpha_2 s$$

where s is an indicator denoting if the agent made a deal in the previous round.

As opposed to the statistical analysis of (10), the residual in the auto-regressive model was modelled as a random noise to make the simulations more realistic. Under this assumption, the LMDA becomes

$$d_i = \alpha_1 d_{i-1} + \alpha_2 s + n_i,$$

where n is a random variable with a zero-centered Laplacian distribution and a scale parameter of 5. This noisy auto-regressive process aims to simulate the deviations in the realized offers from an underlying rational strategy pursued by the realistic agents due to imperfections or unaccounted shifts in the market.

2.3 Price Aggressive Agents

To explore the effects of agents which aggressively attempt to trade even with a low return, the Price Aggressive Agents (PAA) are introduced. Instead of adding s as a parameter in the linear response, the agent uses it as an indicator whether they should be aggressive in their demand. In other words, if the previous round of an agent was successful and s is True, then

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

If the previous round of the agent was unsuccessful and s is False, then

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

where ϵ is the level of aggressiveness of the agent. Therefore, the agent becomes aggressive after an unsuccessful round.

Analogies can be drawn between the PAA and freemium pricing (11) and predatory pricing strategies (12) in market economics. In each case, aggressive price reductions are sought by sellers in order to either increase their market shares or to eliminate incoming competition, even with the risk of short term losses. One such example is monopoly, which can typically employ predatory pricing to eliminate competitions.

2.4 Deep RL Agents

For the deep RL agents, the so-called Deep Deterministic Policy Gradient (DDPG) framework is used for learning. (13) DDPG is an actor-critic deep RL algorithm suitable for RL problems with a continuous action space. Since it is challenging to use Q-value estimations in continuous time, policy gradient methods such as DDPG parametrize the policy of an agent π_θ instead of using a differentiable model such as neural networks. A fully-connected 3-layer neural network is used to model the policy of such an agent. The critic network consists of 4 layers, the actions are appended at the second layer to prevent vanishing gradients and subsequent slow learning.

To briefly describe the learning algorithm, we learn the functions π_θ and Q_θ. The Q-function for a given policy π quantifies the expected time-discounted reward if we perform action a at a state s:

$$Q(s,a) = \mathbb{E}\left[\sum_t \gamma^t r_t | s_0 = s, a_0 = a\right]$$

Learning this function can be achieved by using the Bellman's identity, essentially converting the problem to a regression task. Actor-critic methods aim to learn Q_θ and π_θ simultaneously, and the policy's rewards are gradually increased by back-propagating through the Q.

Environment. In the learning experiments, only one RL agent is trained at a time. The market is then filled with less intelligent buyers and sellers, which can be either ZIA, LMDA or PAA, to allow for comparison between the average earnings of the pool and that of the trained agent.

Implementation and Training. The implementation was done using PyTorch. We train the models for 700 games (each taking 10 rounds), and perform a DDPG iteration after the end of every round (consisting of 10 time steps). We use the Adam optimizer for learning parameters.

Learning stability. To enable numerical stability during learning, an experience memory of 100000 entries composed of the state, the action, the next observed state and the reward was used. A mini-batch of 64 entries was sampled from the experience memory

at each iteration. Ideas from double Q-learning was used, such that a slowly moving target network for determining labels in the Q-learning task of the critic was used. Finally, the deep Q-network (i.e. the critic) was trained for several iterations on a regression task using the experimental dataset before reinforcement learning for stability. Once a stable implementation was found, the weights were used for initializing later experiments.

Exploration vs. exploitation. A recurring theme in reinforcement learning is the trade-off between exploitation and exploration. During the learning process, the RL agent should sufficiently explore different policies and not get stuck at a suboptimal solution. An additive Gaussian exploration model,

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

is employed under which the agent explores a neighbourhood of the current policy proposal. To keep the agent entropy in check and enable learning long-term strategies, σ_i is annealed linearly for the first 300 epochs of the training process from $\sigma_{init} = 50$ to $\sigma_{final} = 3$. Another strategy common in continuous domain RL is using correlated noise for exploration (13). To employ this strategy, an Ornstein-Uhlenbeck (OU) process was also used in the exploration process. OU simulates Brownian motion with friction and can generate correlated noise better suited for long term continuous time exploration.

3 Experiment Results

3.1 Zero-Intelligence Agents

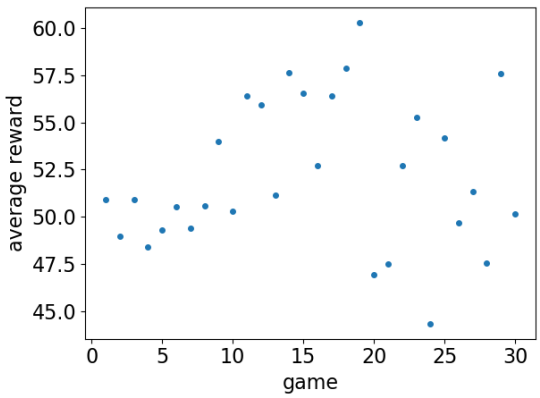


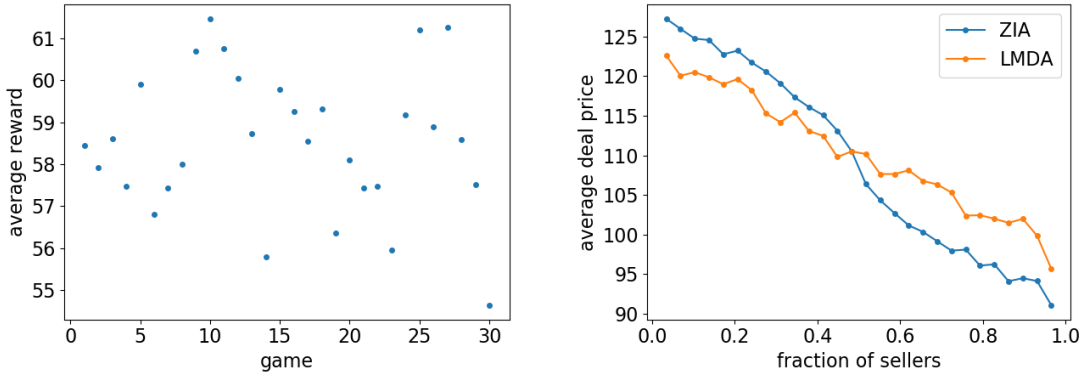
Figure 2: The average reward per game for zero-intelligence agents across 30 games. The number of buyers and sellers were both fixed at 100. As no learning was involved and the generation of the offers relied on random variables, no correlation was shown in the average reward across different games as expected.

Figure 2 shows the average reward per game of a ZIA seller across 30 independent games with 100 buyers and 100 sellers. Since no learning was involved in these agents and the generation of new offers involved random variables, the average reward across different games showed no correlation as expected.

3.2 Linear Markov Decision Agents

The average reward per game of an LMDA seller across 30 independent games with 100 sellers and 100 buyers is shown in Figure 3a. There is still no correlation between the rewards across different games as no learning was implemented, though the average reward for LMDA is found to be higher than that for ZIA.

The average deal price per game for different fraction of sellers is shown in Figure 3b for ZIA and LMDA, with the total number of agents fixed at 500. It is shown that the average deal price decreases with the increasing fraction of sellers for both ZIA and LMDA due to the competition between the sellers. The slope of the graph corresponds the toughness of the competition, and it can be deduced that the ZIA faced a higher competition compared to LMDA, as indicated by the higher rate of decrease in the deal price. The graphs of ZIA and LMDA intersect when there is an equal number of sellers and buyers.

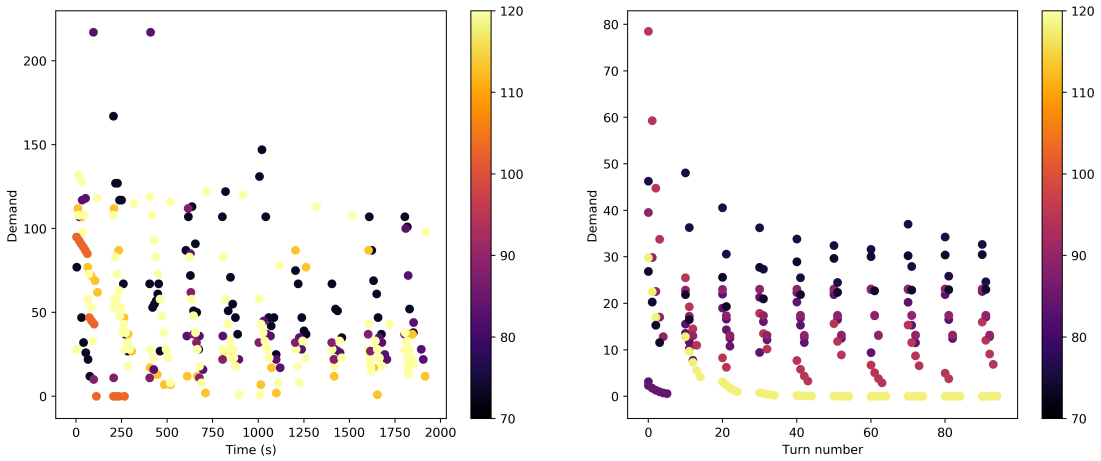


(a) (b)

Figure 3: Left: The average reward per game of a Linear Markov Decision (LMDA) seller across 30 games, each game consisting of 100 rounds. The average reward for the LMDA seller is found to be higher than the ZIA seller shown in Figure 2. Right: Average deal price per game for different fractions of sellers, for the Zero-Intelligence Agents (ZIA) and the Linear Markov Decision Agents (LMDA). The total number of agents were fixed at 500. It is shown that ZIA faced a higher competition compared to LMDA, as indicated by the more rapidly decreasing deal price with the increase in fraction of sellers in the case of ZIA.

Figure 4 shows the demands per round for the experimental data (10) and LMDA, where

the colour scale indicates the reservation price. As expected, both the models naturally converge to higher demands if the reservation price is lower. Even though the experimental data is more noisy, the gradual trend is similar in both plots. In both cases, users with similar reservation prices seem to decrease their demand after unsuccessful offers and increase their demand after successful offers, leading to oscillating demand.



(a) Experimental (data from (10))

(b) LMDA

Figure 4: Experimental demands over time vs. LMDA demands over rounds (time steps).

3.3 Price Aggressive Agents

Figure 5 shows the demands of the PAA at different time steps for different degrees of aggressiveness, indicated by ϵ. The orange plots show the demands of the PAA and the blue plots show the average demands of all other agents in the market, which are of the LMDA type. As the PAA with $\epsilon = 0$ is the same as the LMDA, the demands of both the PAA and LMDA coincide for this case, as shown in Figure 5(a). For $\epsilon = 0.4$, the demands of PAA are lower than those of LMDA at all time steps, and the PAA was found successful in making a deal in a shorter time.

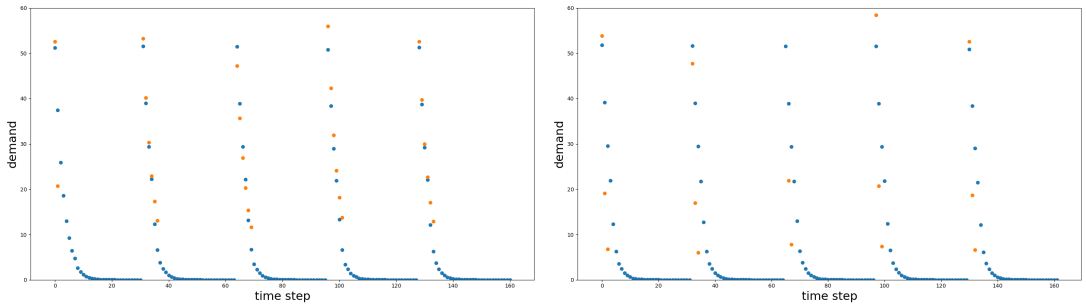
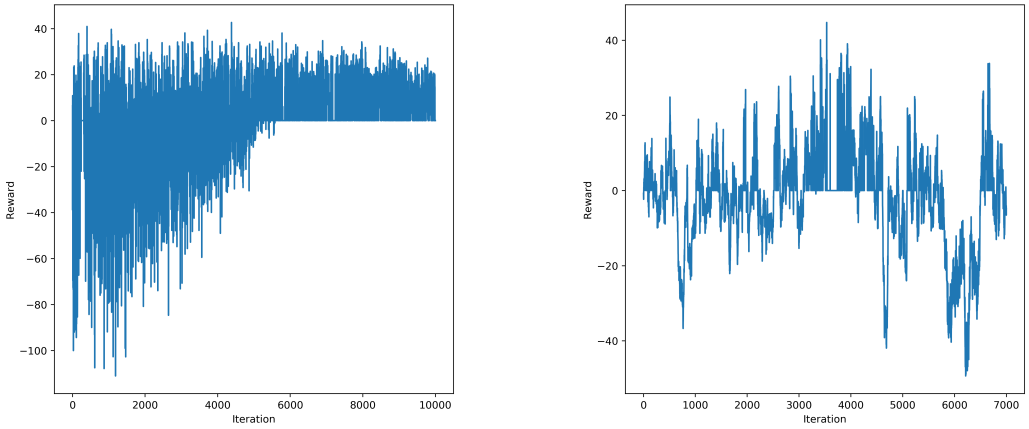


Figure 5: The demands of the PAA at different time steps for different aggressiveness parameters, ϵ. The orange color shows the demands of the PAA, whereas the blue color depicts the average of all other LMDA in the market. (a) PAA with $\epsilon = 0$. In this case, the PAA is the same as the LMDA and so the demands for both the PAA and the LMDA agents coincide. (b) PAA with $\epsilon = 0.4$. It is shown that the PAA has a lower demand than the LMDAs after the initial time step, and makes a deal in a shorter time.

3.4 Deep Reinforcement Learning Model

In this section we outline the results for the RL learning algorithm. Firstly, in Figure 6, we have the evolution of the agent's learning process over training iteration for the two different exploration algorithms, Gaussian exploration policy and Ornstein-Uhlenbeck Process (OU).

One can observe that as expected, even though the rewards are negative at early stages of learning, the agent quickly learns to avoid negative demands. As expected, even though not directly modeled with a nonzero non-linearity due to numerical issues, the learning algorithm quickly forces the policy network to have positive demand only. The final episodes typically tend to have much higher rewards. The OU exploration policy is however more prone to get stuck in negative reward regions. This however does not affect performance, probably because correlated positive noise is much more sample efficient for exploration in this scenario. The performance of each RLA model with different types of agent pools is shown in Table 1. Overall, the RL agent manages to earn significantly more than any other model. One can also see that the RL agent has difficulty competing against aggressive pricing when there are more buyers than sellers. This is because in this scenario the profit margin the RL agent has to compete against is small, forcing the returns to be significantly lower.



(a) Gaussian exploration policy

(b) OU exploration policy

Figure 6: Rewards over iterations for RL policies with different learning strategies. For the Gaussian exploration policy, the agent quickly learned to avoid the negative reward region, resulting in only positive rewards at a later time step. In the case of OU exploration policy, the agent was more prone to being trapped in the negative reward region.

Table 1: Rewards of the RLA agents against different pools of agents. We also indicate the exploration strategy employed.

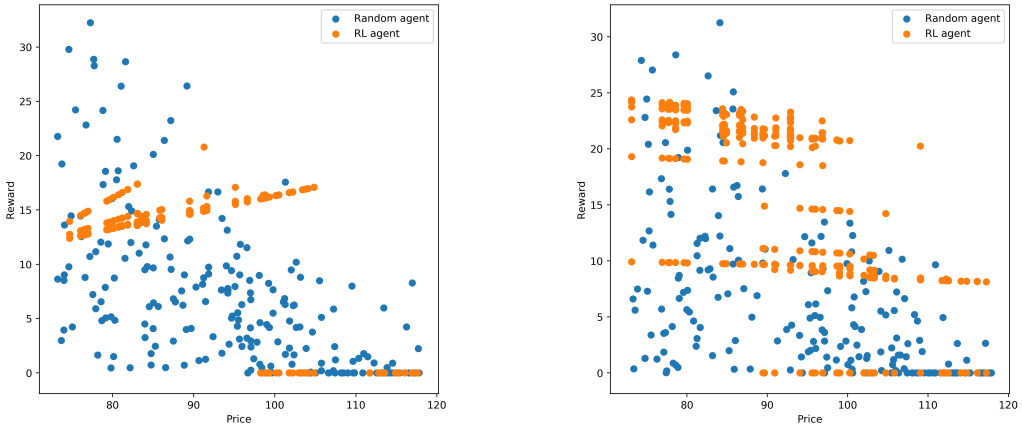
Agent Pool	Pool Earnings	RLA Earnings	Learning Agent
ZIA	5.23	9.27	RLA+Gaussian
ZIA	7.01	8.39	RLA+OU
ZIA	5.27	12.32	RLA+OU+anneal
LMDA	7.19	10.46	RLA+Gaussian
LMDA	7.42	10.72	RLA+OU
PAA	3.71	6.91	RLA+OU

3.5 Interpreting the Behaviour of the Deep Model

The Deep RL model clearly is superior to the competing agents in terms of performance. However, the deep policy network we train has the weakness that it is a completely blackbox agent. Even though it performs well, its rationale and strategy can not be understood directly. In this section, we make several empirical observations about the policy network via agent based simulations.

Most importantly, we plot the reservation price vs. average earnings of zero intelligence

agents and the RLA are plotted in Figure 7. As expected the reservation prices affect the profit margin the agent aims for. Interestingly, although the RLA+OU agent has reasonably concluded that making smaller profits for higher reservation prices is better, the RLA+Gaussian agent has a larger profit margin for greater prices. There might be two competing objectives in the case of determining the demand of for a fixed reservation price. Firstly, it is difficult to sell for a higher reservation price as the market is likely to reject. However, at higher prices simpler models tend to perform worse, therefore, the RLA might have discovered that it is meaningful to compete directly. However, at higher prices, the agent still can fail to conclude deals. This can be due to insufficient exploration of this region, or because the agent does not receive an explicit negative reward penalty for not concluding a deal at all. With the effect of a discount factor of $\gamma = 0.97$, the agent might not be sufficiently incentivized to conclude a deal.



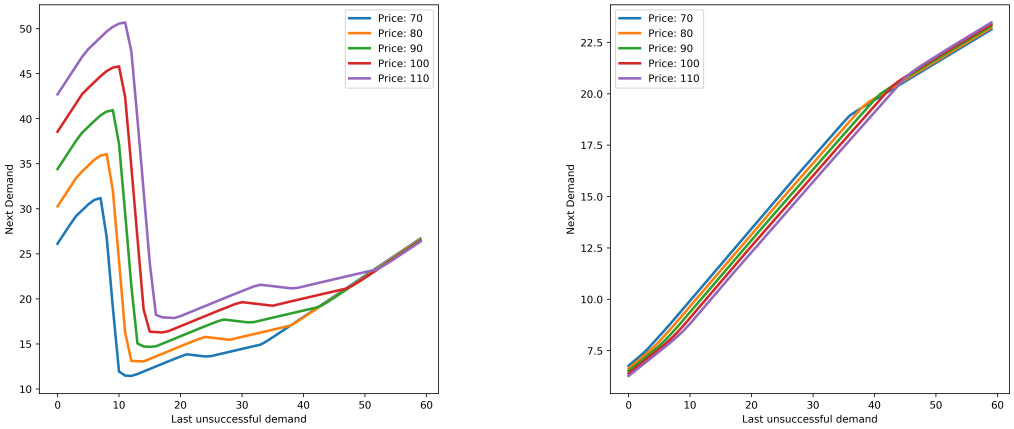
(a) Gaussian exploration policy

(b) OU exploration policy

Figure 7: The reward vs. reservation prices for the RL agents.

In Figure 8, one can see the plots of the agents "modified" offer after an unsuccessful one. For the RLA Gaussian agent, there seems to be a transition point where the model decides between an ambitious offer for the next time step and a safer offer. This might be an artefact of the trade-off between a safe demand and higher returns. Namely, if the last offer was low enough, the agent finds greater expected returns for a much higher price. In fact, as the game processes, other agents will also play more safely, leaving room for higher profits. Moreover, for higher previous offers, the effect of the agent's reservation price seems to diminish. However, as the test time rewards of the OU agent is much better, the behavioural pattern of the Gaussian-exploring agent might simply be due to

poor exploration of the policy space. Finally, note that in the case of the OU agent a greater reservation price corresponds to a less ambitious demand for the next time step as the "competitiveness" of the product decreases.

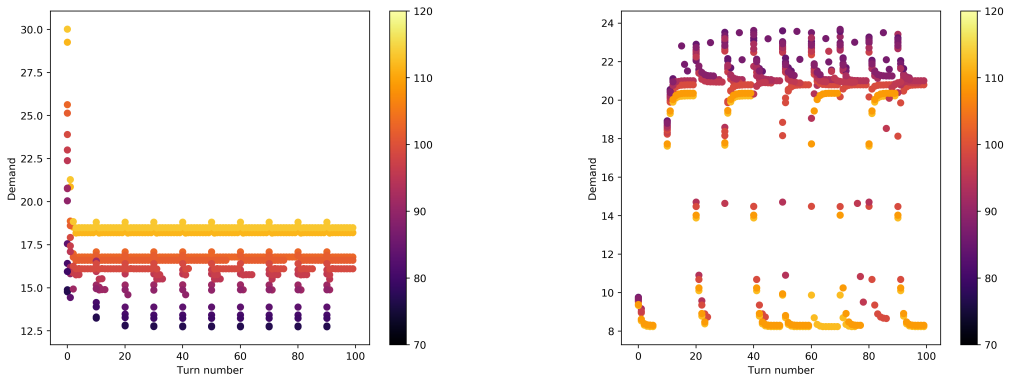


(a) Gaussian exploration policy

(b) OU exploration policy

Figure 8: How the RL models choose to place their next offer in the game, given their previous offer was unsuccessful and did not lead to a deal.

Finally, we also plot the demand values of RL models in time, sampling over several games and reservation prices in Figure 9. Firstly, one can observe that each agent has to decrease their demand after an unsuccessful attempt, whereas they can increase it after a successful attempt. This is in line with expectations, and also in line with experimental data. However, it seems to be the case that humans are usually much more conservative the the RLA, whereas the RLA can demand higher profit margins more confidently. The RLA trained with the OU agent seems to have learned that lower reservation prices give more room for speculation, and the RLA can be more confidently demand a higher "profit margin" from the market as its production is competitive.



(a) Gaussian exploration policy

(b) OU exploration policy

Figure 9: Sample games with the demands of the RLA at each round plotted over time. Since we have 10 rounds with 10 time steps each, the plots are over 100 time steps. Different colours belong to different agents, and the shade of the colour indicates the resevation price of that agent.

4 Conclusion

It was found that the agents trained using Reinforcement Learning outperformed the agent pool consisting of ZIA, LMDA and PAA. It was also found that the rewards for the RL agent trained using the Gaussian exploration policy increased with the number of iterations, with the agent learning to avoid negative rewards. Such an agent has a larger profit margin for greater reservation prices, while those trained under the OU policy has reasonably concluded that making smaller profits for higher reservation prices is better. In conclusion, it was found that humans are usually much more conservative compared to the the RLA, whereas the RLA can demand higher profit margins more confidently. The RLA trained with the OU agent seems to have learned that lower reservation prices give more room for speculation, and the RLA can be more confidently demand a higher "profit margin" from the market as its production is competitive. An interesting extension to this work would be the inclusion of the additional information into the observation space of the intelligent agent to model more realistic scenarios.

References

[1] S. Lim, *Auction Market*, 2019 (accessed December 6, 2019).

[2] *New York Stock Exchange*, (accessed December 6, 2019).

[3] D. Pandey and P. Pandey, "Approximate q-learning: An introduction," in *2010 second international conference on machine learning and computing*, pp. 317–320, IEEE, 2010.

[4] L. Ma and Y. Liu, "Application of a deep reinforcement learning method in financial market trading," in *2019 11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, pp. 421–425, IEEE, 2019.

[5] V. Gullapalli, "A comparison of supervised and reinforcement learning methods on a reinforcement learning task," in *Proceedings of the 1991 IEEE International Symposium on Intelligent Control*, pp. 394–399, IEEE, 1991.

[6] T. L. Meng and M. Khushi, "Reinforcement learning in financial markets," 20 July 2019.

[7] A. V. Rutkauskas and T. Ramanauskas, "Building an artificial stock market populated by reinforcement-learning agents," *Journal of Business Economics and Management*, September 2009.

[8] J. Carapuço, R. Neves, and N. Horta, "Reinforcement learning applied to forex trading," *Applied Soft Computing*, vol. 73, pp. 783–794, 2018.

[9] T. Asikis, *ABMSSS Project Script*, 2019.

[10] H. Nax, D. Nunez-Duran, B. Pradelski, and E. COSS, "Feedback effects in the experimental double auction with private information,"

[11] D. Semenzin, E. Meulendijks, W. Seele, C. Wagner, and S. Brinkkemper, "Differentiation in freemium: Where does the line lie?," in *Lecture Notes in Business Information Processing*, pp. 394–399, Springer, 2012.

[12] O. E. Williamson, "Predatory pricing: A strategic and welfare analysis," *Expert Systems with Applications*, vol. 87, p. 284–340, 1977.

[13] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

5 Appendix

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from GitHub at `https://github.com/Alehud/GESS`. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Batuhan Yardim, Aleksei Khudorozhkov, Neri Passaleva, Ka Rin Sim

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

DEEP REINFORCEMENT LEARNING FOR DOUBLE AUCTION PROCESSES

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Yardim	Batuhan
Khudorozhkov	Alexsei
Passaleva	Neri
Sim	Ka Rin

With my signature I confirm that

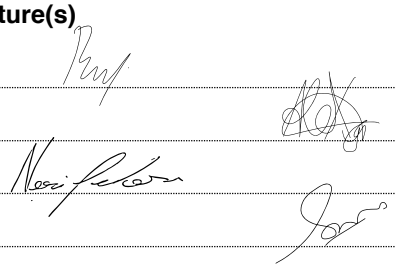
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date

Zürich 06/12/2019

Signature(s)



For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.