COMPUTER RESEARCH AND MODELING
2024 VOL. 16 NO. 7 P. 1793–1812
DOI: 10.20537/2076-7633-2024-16-7-1793-1812

Ки&М

SPECIAL ISSUE

UDC: 519.8

Reinforcement learning in optimisation of financial market trading strategy parameters

R. L. Vetrin[a], K. Koberg[b]

Innopolis University,
1 Universitetskaya st., Innopolis, 420500, Russia

E-mail: [a] r.vetrin@innopolis.university, [b] k.kaberg@innopolis.university

Received 29.10.2024, after completion — 15.11.2024
Accepted for publication 25.11.2024

High frequency algorithmic trading became is a subclass of trading which is focused on gaining basis-point like profitability on sub-second time frames. Such trading strategies do not depend on most of the factors eligible for the longer-term trading and require specific approach. There were many attempts to utilize machine learning techniques to both high and low frequency trading. However, it is still having limited application in the real world trading due to high exposure to overfitting, requirements for rapid adaptation to new market regimes and overall instability of the results. We conducted a comprehensive research on combination of known quantitative theory and reinforcement learning methods in order derive more effective and robust approach at construction of automated trading system in an attempt to create a support for a known algorithmic trading techniques. Using classical price behavior theories as well as modern application cases in sub-millisecond trading, we utilized the Reinforcement Learning models in order to improve quality of the algorithms. As a result, we derived a robust model which utilize Deep Reinforcement learning in order to optimise static market making trading algorithms' parameters capable of online learning on live data. More specifically, we explored the system in the derivatives cryptocurrency market which mostly not dependent on external factors in short terms. Our research was implemented in high-frequency environment and the final models showed capability to operate within accepted high-frequency trading time-frames. We compared various combinations of Deep Reinforcement Learning approaches and the classic algorithms and evaluated robustness and effectiveness of improvements for each combination.

Keywords: deep reinforcement learning, algorithmic trading, high-frequency trading, market making

Citation: *Computer Research and Modeling*, 2024, vol. 16, no. 7, pp. 1793–1812.

КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2024 Т. 16 № 7 С. 1793–1812
DOI: 10.20537/2076-7633-2024-16-7-1793-1812

Ки&М

СПЕЦИАЛЬНЫЙ ВЫПУСК

УДК: 519.8

Обучение с подкреплением при оптимизации параметров торговой стратегии на финансовых рынках

Р. Л. Ветрин[a], К. Коберг[b]

Университет Иннополис,
Россия, 420500, г. Иннополис, ул. Университетская, д. 1

E-mail: [a] r.vetrin@innopolis.university, [b] k.kaberg@innopolis.university

Получено 29.10.2024, после доработки — 15.11.2024
Принято к публикации 25.11.2024

Высокочастотная алгоритмическая торговля — это подкласс трейдинга, ориентированный на получение прибыли на субсекундных временных интервалах. Такие торговые стратегии не зависят от большинства факторов, подходящих для долгосрочной торговли, и требуют особого подхода. Было много попыток использовать методы машинного обучения как для высоко-, так и для низкочастотной торговли. Однако они по-прежнему имеют ограниченное применение на практике из-за высокой подверженности переобучению, требований к быстрой адаптации к новым режимам рынка и общей нестабильности результатов. Мы провели комплексное исследование по сочетанию известных количественных теорий и методов обучения с подкреплением, чтобы вывести более эффективный и надежный подход при построении автоматизированной торговой системы в попытке создать поддержку для известных алгоритмических торговых техник. Используя классические теории поведения цен, а также современные примеры применения в субмиллисекундной торговле, мы применили модели обучения с усилением для улучшения качества алгоритмов. В результате мы создали надежную модель, использующую глубокое обучение с усилением для оптимизации параметров статических торговых алгоритмов, способных к онлайн-обучению на живых данных. Более конкретно, мы исследовали систему на срочном криптовалютном рынке, который в основном не зависит от внешних факторов в краткосрочной перспективе. Наше исследование было реализовано в высокочастотной среде, и итоговые модели показали способность работать в рамках принятых таймфреймов высокочастотной торговли. Мы сравнили различные комбинации подходов глубинного обучения с подкреплением и классических алгоритмов и оценили устойчивость и эффективность улучшений для каждой комбинации.

Ключевые слова: обучение с подкреплением, алгоритмическая торговля, высокочастотная торговля, маркет-мейкинг

Nomenclature

AS	Avellaneda – Stoikov
BTC	Bitcoin
DDQN	Double Deep Q-network
DDPG	Deep Deterministic Policy Gradient
DPG	Deterministic Policy Gradient
DQN	Deep Q-network
DRL	Deep Reinforcement Learning
ML	Machine Learning
ODE	Stochastic differential equation
OU	Ornstein – Uhlenbeck
PnL	Profit and Loss
PPO	Proximal Policy Optimisation
RL	Reinforcement Learning
TD3	Twin Delayed Deep Deterministic Policy Gradient
USD	United States Dollar
USDT	Tether United States Dollar

Introduction

A large number of papers devoted to construction and optimisation of trading strategies. First widely-recognized theories on price trend estimations can be found back in 1900 when Louis Bachelier [Bachelier, 1900] for the first time described stochastic approach at price prediction. Since then theory was extensively expanded by works of Paul Levy [Levi, 2017], Alexander Hinchin [Hinchin, 1936] and Kiyosi Ito [Itô, 1951] and other researchers forming a foundation of modern modeling of price predictions.

Meanwhile the speed of trading was constantly rising up from manual enter of orders in the Wall Street Exchange pit to electronic execution with sub-millisecond execution. This trend opened up whole new markets of high-frequency trading which is nearly independent from long-term tradings.

As the same time, since mid 2000, computing technology has drastically improved pawning the way for ML techniques which were theorised during 20th century [Cartea, Jaimungal, Penalva, 2015]. Since high-frequency trading requires highly-optimised hardware and software to preserve low latency order composition, ML was not widely implemented into such strategies. However, with recent hardware development, high-frequency ML have become a viable option.

In this project we will explore application of Reinforcement Learning techniques in high frequency algorithmic trading for optimisation of existing market making strategies. We will explore how traditional trading approaches can be linked with RL terms and how such techniques can help to converge trading problem. In particular, we will demonstrate application of popular RL algorithms improving static models' parameters maximising the overall returns in high-frequency environment.

In Section "Related work" we will explore existing static and RL solutions in financial trading. In Section "Methodology" we will formulate core concepts of purposed models based on principles and methods described in the latter section. In the next two sections we will describe our experimentation set-up as well as discuss results of those. Finally, in the last Section we will summarise our work done and draw some conclusions.

Related work

In this section we will explore known approaches to RL in high-frequency trading as well as those of quantitative algorithms. We will establish current approaches and possible modifications of those.

Trading exchange

Term “Exchange” refers to a marketplace where commodities, derivatives and other financial instruments are traded [Cartea, Jaimungal, Penalva, 2015]. With development of new commodities (such as cryptocurrencies) has widened repertoire of financial instruments (such as perpetual derivatives), new exchanges and markets has emerged. In addition, with development of trading infrastructure, a whole new segment was established — high frequency trading with execution speed of under 500 microseconds.

With development of theory of market prices, various financial instruments came to place. Among the newest most influential derivatives suggested are perpetual futures first suggested by Robert Shiller [Shiller, 1993]. As the name suggests, such futures do not have an expiration date, meaning that those can be held indefinitely without the need to re-roll the contracts. However, the such instrument did not get too much attention up until 2011 when Alexey Bragin [Forbes Russia, 2014] introduced it for cryptocurrencies on ICBIT exchange. This allowed to enlarge leverage trading for cryptocurrencies and attracted new capital to cryptocurrency trading. As of March 2024, average daily volume traded of spot BTC/USDT (Bitcoin against USD Tether) totaled approx. 2 billion USD against 16 billion USD of BTC/USDT perpetual futures.

Since our research will be focused on technical analysis aspects, we will conduct our research within short-term frequencies on cryptocurrencies, minimising relation of sentiment or other non-technical factors. More specifically, we will focus on perpetual futures market since the market is more fluid and thus more suitable for known statistical methods. Finally, we will be using data from Binance exchange since it is rated as the largest cryptocurrency exchange as of March 2024 [CoinMarketCap].

High frequency algorithmic trading

There are two most popular high-frequency strategy classes: market making and statistical arbitrage.

Market making strategy suggests calculation of optimal ask and bid quotes around current mid point aiming to execute both, receiving profit in amount of spread between quoted prices (with consideration of commission). Large exchanges offer negative commissions to or hire market-making traders in order to stimulate them to improve liquidity. However, without simulations, it proved to be challenging to fit profitable statistical market making model. Many market making strategies involve statistical arbitrage techniques which employ mean reversion models calculated basing on synthetic features [Lo, 2010].

Latency/triangle arbitrage strategy relies on price latency among different securities and exchanges. Upon conversion of those securities from to another and converting it back to the domain asset, there can be guaranteed profit caused by price latency. Advanced algorithms on statistical arbitrage can use more than dozens of conversions. Naturally, high execution speed is the key for this strategy since a window for execution is slim [Wah, Wellman, 2013].

There are other mainstream strategies such as quote stuffing or spoofing [Lee, Eom, Park, 2013]. However, those strategies largely imply manipulation of posted orders which are not meant to execute. Thus, modern exchanges often take measures against traders who utilise such strategies since it increases pressure in exchange’s infrastructure and risk management with little real trades outcome.

Since research of ML methods in statistical arbitrage would require very delicate simulation of various exchanges and price trends as well as signal latency among them, we will primarily focus on market making strategies class during our work.

Quantitative algorithms

Most of the quantitative algorithms rely on description of the price process. One of the most popular processes used in description of price trends is the OU process [Uhlenbeck, Ornstein, 1930]. This model and it's variations were thoroughly investigated and promoted by many authors [Butler, King, 2004; Hansen, Pienaar, Orzack, 2008; Beaulieu et al., 2012] both from financial and natural field. The process has the following form:

$$X(t) = \mu + e^{-at}\left(X(0) - \mu + \sigma \int_0^t e^{as}\, dB(s)\right), \tag{1}$$

where $X(t)$ resembles price at time t, μ is a constant and mean at stationary, $B(s)$ is Brownian motion at time and α is described as strength of a resisting force to stabilizing sections. According to known properties of OU, it has a Gaussian distribution of $N\left(\mu + e^{-\alpha t}, \frac{\sigma^2}{2\alpha\left(1-e^{2\alpha t}\right)}\right)$. Note that with $t \to \infty$, exponential terms converge to zero. In addition, it is the only process that satisfy the three properties — Markov, Gaussian and stationary. For one-dimensional OU process, the SDE defined as follows:

$$dX(t) = \alpha(X_t - \mu)dt + \sigma\, dW_t. \tag{2}$$

Such model offer us a stationary process estimation considering there is no distinctive trend in place. This leads us to a non-stationary form of OU model:

$$dX(t) = (\beta + \alpha(\beta t + \gamma - X_t))\, dt + \sigma dW_t, \tag{3}$$

where $\beta t + \gamma$ referred to as "short-term deterministic fair price" [Merjin, Averbuch, 2024]. Notably, such process known to have close to zero values of α after calibration process which leads to rejection of hypothesis of underlying fair price.

There are a lot of OU modifications and approaches to calibrate the parameters of the model. For instance, there is skew OU process [Wang, Song, Wang, 2015] which adds the $L_t(0)$ symmetric local time of X to the classic OU SDE formulated at equation (2). A robust approach to evaluate the parameters were introduced by Rieder [Rieder, 2012] and consist of evaluation of the three parameters:

$$\mu = \overline{X}, \quad \alpha = -\frac{\ln(\rho)}{d}, \quad \sigma^2 = 2\alpha s^2, \tag{4}$$

where ρ is the auto-correlation at lag 1 and s^2 is the variance. More recent approach suggest to improve the calculation by introduction of Monte Carlo method and root finding algorithm for better approximation of auto-correlation estimator [Kramer, 2022]. Another recent approach suggest considering α as a separate OU process [Merjin, Averbuch, 2024]. Another important modification of OU process is the OU process with jumps [Priola, Zabczyk, 2009]. The related research indicate that occasional jumps in pricing process may affect the classic OU parameters.

Another approach of price quotation was introduced by Avellaneda and Stoikov [Avellaneda, Stoikov, 2008]. In their research, they presented a model of quotes taking into consideration current inventory and probability of new trades arrival. Specifically, paper outlines the reservation price $r(s, t)$:

$$r(s, t) = s - q\gamma\sigma^2(T - t), \tag{5}$$

which account the current midprice s for the stock q currently held by the agent at time t using a constant γ. Around that price there is a proposed spread of ask and bid quotes:

$$\delta = \gamma\sigma^2(T - t) + \frac{2}{\gamma}\ln\left(1 + \frac{\gamma}{k}\right), \tag{6}$$

where k is a constant. The approach suggest quoting the prices considering risk of having an inventory and measured distribution of pricing shift. An approach of similar direction was taken by Guilbaud and Pham [Guilbaud, Pham, 2013]. That work have introduced regime switching control utility as well as doubly stochastic Poisson process for determination of ask/bid shift expectations.

Another algorithm suggests usage of volume imbalance among first levels or order book [Amir, Masoud, 2020]. It was showed in experiments that imbalance of volume in the first levels of order book leads to short-term indication of a price shift. Traditional order book imbalance can be formulated as follows [Cartea, Donnelly, Jaimungal, 2018]:

$$imb = \frac{V_l^b - V_l^a}{V_l^b + V_l^a}, \tag{7}$$

where V is a cumulative volume up to level l for ask (a) or bid (b) side.

A combination of "grid" strategy and ML technique was purposed for fiat market [Rundo et al., 2019]. The method included a "Regression Network" for calculation of closing price for a certain short-term period. Then a Trend Classification block classifies a trend as "Long", "Short" or "Null" using approximation of first and second derivative of newly obtain price trends. Using that trend classification, following blocks apply basic risk management constrains and output quotes in case of non-"Null" trend.

Reinforcement learning approaches

Reinforcement learning is a subcategory of Machine learning which primarily distinct by the trial-and-error approach to learning rather than classic supervising learning. On each iteration an RL model receives a state from an environment, sends a designated action to an environment and collect a feedback in form of reward and new state. Aim of RL algorithm is to formulate optimal policy which maximizes expected future reward.

In order to solve above mentioned task, RL approach suggests modeling of a Markov decision processes (MDPs) which consists of a set of states, a set of actions, a reward function and state transition functions [Kaelbling, Littman, Moore, 1996]. By evaluation of various pairs of states and actions, we may formulate a value function as below:

$$V^*(s) = \max_a \left(R(s, a) + \gamma \sum T(s, a, s')V^*(s')\right), \tag{8}$$

where $R(s, a)$ is a reward function, $T(s, a, s')$ is a transition among states s and s' with action a. That leads us to the optimal policy:

$$\pi^*(s) = \arg\max_a \left(R(s, a) + \gamma \sum T(s, a, s')V^*(s')\right). \tag{9}$$

One way, to find an optimal policy is to utilize sample backup by continuous update of value for a certain state-action pair:

$$Q(s, a) = Q(s, a) + \alpha\left(R(s, a) + \gamma \max_{a'} Q(s', a') - Q(s, a)\right), \tag{10}$$

where $Q(s, a)$ is usually refereed to as the Q-value for some state-action pair [Singh, 1993].

Naturally, in a real-world scenario, transition matrix is inconceivable to record and maintain. In order to overcome this limitation, it was purposed to combine neural network approach with concepts of RL. DRL [Mnih et al., 2015] approach suggest to replace Q-value calculations by a neural network. Having target Q-value given by Eq. (10) and current Q-value given by a neural network, we may calculate an error and optimize our Deep Q-Network.

As discussed earlier, aim of RL methods is to derive optimal agent policy utilising accurate estimator of Q-value dependent on state-action pair. One the most obvious (from ML perspective) ways to achieve that is Deep Q-network [Mnih et al., 2015] idea of which we described earlier. However, it was quickly noted that stability of the model can be improved by adding target neural net composing the DDQN [Van Hasselt, Guez, Silver, 2016]. By adding a delayed-updated target network for Q-value calculations, Hasselt, et. al were able to counter overestimation encountered in DQN network. More recent modification of DDQN was achieved by Li et al. [Li et al., 2022] who introduced a Weighted DDQN. This approach suggests more gradual update of target network by applying weight coefficient:

$$\theta^{tar'} = \beta \cdot \theta^{tar} + (1 - \beta)\theta^{pred}, \tag{11}$$

where weights of a target network θ^{tar} get updated by prediction network θ^{pred} using a constant β. According to the results, such approach helped to fine-tune trade-off between overestimation and underestimation. Since the model relies on evaluation of action-state pair, DQN and DDQN are only able to work in discrete action space.

In order to enlarge possibilities of deep RL methods and transfer action space into continuous space, an actor-critique approach was developed [Sutton, Barto, 2018] which quickly evolved into DPG algorithm [Silver et al., 2014]. The approach suggest to learn a value function (aka "Critic") and policy (aka "Actor"). Actor maps incoming state into an action and critic responsible for evaluation of the actions taken by an actor. By extending DPG with the DDQN and other ideas, we received DDPG [Casas, 2017] and multiple modifications of that [Zhang et al., 2021; Wu et al., 2020; Dong, Yu, Ge, 2020]. DDPG combines previously discussed features like target networks for actor and critic. The next improvement came with additional pair critic-target critic with a TD3 [Fujimoto, van Hoof, Meger, 2018] model. The model has two sets of critic models (both have a target counterpart) both of which take part in temporal difference calculation. Minimum difference of those is taken for new Q-value calculation. Additionally, gradient clipping and models weights regularisation was added as means of overestimation/underestimation control.

More recent popular algorithm is a PPO [Schulman et al., 2017] which takes advantages of the Trust Region Policy Optimization algorithm [Schulman et al., 2015]. The algorithm considers abstraction of policy network and value network. A policy network represents a stochastic policy which updates using entropy error of a new policy compared to older version. Value network evaluates "advantage" of current state and optimises by an error compared to sum of discounted rewards. By using combination of an error and clipping of the old/new policy coefficients, policy network gradients are calculating with respect to the value network. This model can be implemented in actor-critic style with treatment of soft-maxed policy network outputs as action probability. However, this leads to limitations of action space to a discrete space. Another approach suggest treatment of policy network's outputs and parameters of a distribution (usually, a of a normal distribution). This approach allows to persist continuous action space.

Reinforcement learning in high-frequency trading

Application of RL in finance have recently drawn a lot of attention having sharp increase in publications during 2021–2024. After demonstration of RL capabilities in robotics and games, it seems natural to try and apply the same techniques on financial trading. However, In the early works on RL in finance a number of disadvantages were noted. Specifically, experiments showed that

majority of popular RL methods yield extremely unstable results easily jeopardised by random factors tweaking [Henderson et al., 2018]. Another article suggests that application of RL algorithms face large challenge of reproducibility issue as well as the fact that researchers usually construct over-simplified baselines [Chaouki et al., 2020].

First, we will discuss a problem statement as per the related work. One problem statement was formulated as a management of a portfolio vector at start of a day [Conegundes, Pereira, 2020] meaning that action space represents a portfolio configuration. Similar work proposes to allocate an asset in accordance with Markowitz allocation which considered portfolio risk [Chaouki et al., 2020]. Another approach suggest to simply take a decision at any time point on which assets should be acquired or sold based on observable state which includes current balance, portfolio, etc. [Singh et al., 2022]. There was an attempt to utilize PPO algorithm for optimization of Avellaneda – Stoikov market-making algorithm [Falces Marin, Díaz Pardo de Vera, Lopez Gonzalo, 2022]. Another attempt at improving an existing model was purposed by Yitao Qiu et al. [Qiu, Liu, Lee, 2024]: utilizing DDPG algorithm, authors purposed derivation of Quantum pricing levels in accordance to Quantum Finance Theory. In addition, the PG algorithm was added acting as risk-control agent which select quantum price support or resistance level. In majority of work analyzed, reward was calculated using PnL or similar metric (e. g. Sharpe[1] or Sortino[2]). Another approach was presented by Lussange et al. [Lussange et al., 2021] considering simulation of a exchange by synthetic reinforcement learning agents. Overall we may observe that majority of researchers treat problem of optimal financial trading as a typical RL problem having simulation of an exchange as environment and bid/ask quotes (or a set if binary signals) as an actions.

Now, we will summarize on what methods and additional data were used for trading systems. Many researchers were utilizing DDPG algorithm to solve the problem [Chaouki et al., 2020; Wu et al., 2019; Conegundes, Pereira, 2020]. According to the results, this approach profitable results up to 120 % per year. However, results proved to by highly volatile having notable gaps of drawdowns in contrast to established baselines which experienced lower volatility in PnL dynamics. Another popular model in place are DDQN and DQN models. Main distinction of this approach is a limitation of discrete action space. To counter that, some researchers [Carbonneau, 2021; Liu et al., 2023a] converted quotes into discrete spaces from a midprice value. Another researchers [Carta et al., 2021] considered a three-value action space representing commands “Long”, “Short” and “Hold”. Comparing to the other methods, DQN achieves lower metrics than other model counterparts (however, it is important to note that every experiment was conducted under various conditions and data thus streamline comparison is a subject to a bias). Finally, some researchers attempted more sophisticated methods like TD3 [Kabbani, Duman, 2022] and PPO [Liu et al., 2023b]. The experiments yielded highly profitable results utilizing solely candle stick data(data containing open, close, maximum and minimum price across fixed interval). For experiments on stock data [Kabbani, Duman, 2022] a aggregated sentiment analysis was successfully implemented. Another notable result was achieved by Yuling Huang et al. [Huang et al., 2024] introducing dual DQN each using different input data: TimesNet (a recent approach to transform a time-series from 1D into 2D space) and Multi-Scale Convolution Network transformation of finance data.

Methodology

In later sections we noted few known approaches to Reinforcement learning in algorithmic trading. We may observe that very little of the work reviewed was focused on high-frequency data. Most of the results were conducted on daily market ticks and were not considering cryptocurrencies as

[1] Ratio of risk-free return (a return minus risk-free return ratio. Risk free ration can be a return of government bonds) to standard deviation of that.

[2] Similar metric as Sharpe but instead of risk-free ratio, a “required” ratio is applied and downside standart deviation is used.

the base asset. In our experiments we will aim to extend the results of related work on high-frequency cryptocurrency market and on additional set of underlying market-making models.

Our approach will attempt to combine known static algorithms with RL optimisation of parameters. In addition, we aim at models which can perform in high-frequency environment.

We consider the following role of RL model as parameter optimiser: the static model tends to transform certain indicators into quote decisions. For instance, the OU models tends to analyse auto-correlation among the data as well as variance for a period of time. Thus, another indicators are inaccessible to static methods by design. On contrary, RL models are able to interpret any amount of additional data. Thus, an RL model will be adjusting certain parameters of a static model and this way will add analysed indicators to the static model's decision process.

Environment composition

As stated earlier, we will be using high-frequency data from Binance exchange. Specifically, we will be using BTC/USDT order book snapshots taken every 200 milliseconds. In total we gathered data for the period of 01 April – 30 April with total of 12.9 mln. rows. We will devote data for the period of 01–15 of April for training set and 16–30 of April for test set.

As suggest related research on RL models on order book snapshots [Falces Marin, Díaz Pardo de Vera, Lopez Gonzalo, 2022], information about prices and volume of deep level do not yield any useful information for an RL agent. Thus, for our state space, we restricted number of levels to three as those noted to have most usable information about current volume state. Having a set of current prices and volumes represent a state at a particular point in time. However, as previously stated, future price trends heavily depend on underlying price dynamics. By design, deep RL models achieve this by having constant optimisation using replay memory routines [Silver et al., 2014]. But in order to help our system to track time-series aspect of data, we will be adding technical indicators which represent summary on previous window of ticks.

Each N ticks our model will receive a state and send actions which will be converted to a simulated order. Depending on price dynamics, order may increase or decrease available quantity or simply expire without having an effect. At all times we will be tracking realised PnL, meaning that current inventory will be always converted at current midprice. Model will have information regarding current volume and balance.

We consider possibility of an online learning, each N forward passes we will be optimising the model completing the episode. Thus, parameter optimisation will persist both on training and evaluation runs. Overall high level composition of our model will have the following layout shown in Fig. 1, where the Static Algorithm block can be simply converter from discrete or continuous actions to quotes for orders.

Analysis of volume trend

Earlier we considered a volume imbalance indicator which suggested to be of short-term influence on the future price trends. We believe that this idea can be extended to trend of volume of the first levels of an order book. Even if the price itself is static, changes in volume should be preceded before a price shift. Basing on that, we formulate a following proposition.

Let us have a sequence of sum values of the first N ask or bid levels of an order book across some window of w equally separated ticks. Than we propose to fit a set of coefficients b and ψ such that equation:

$$y_{pred} = A\psi + b, \tag{12}$$

where y_{pred} is a linear regression predictions with A trend sequence $[1, 2, \ldots, w]$. This will give us a least-squares solution for first N levels of an order book ticks volume for our fixed window w. The coefficient ψ in this equation takes a role of current volume change trend for ask or bid side. We

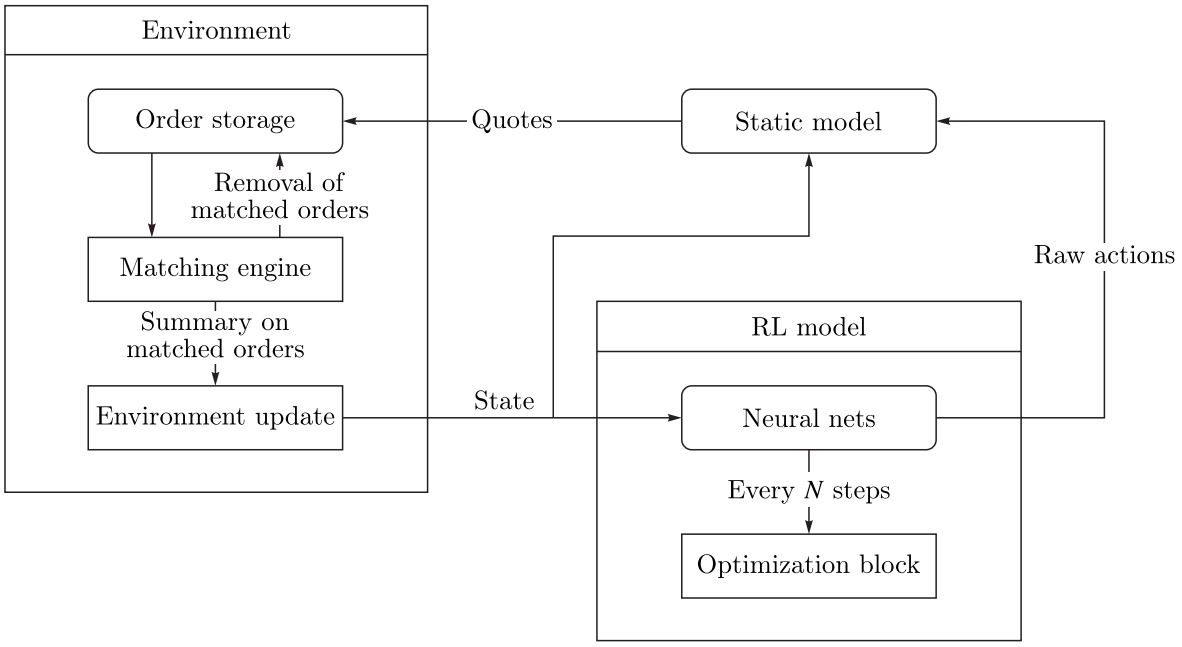

Figure 1. Overview of combined RL and Static model for trading

propose that if coefficients ψ are of different signs for ask and bid size and an absolute value of ψ of both sides is larger than a certain threshold, an indication of a short term trend occurs. Specifically,

$$\alpha = \begin{cases} 1, & \text{if } -\psi^{ask} < -\omega \text{ and } \psi^{bid} > thr, \\ -1, & \text{if } \psi^{ask} > \omega \text{ and } -\psi^{bid} < -thr, \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

where α is a three-value operator which classifies a trend as [1, −1, 0] for positive, negative and null trend respectively and ω is a hyper-parameter.

In order to check the hypothesis, we composed a following experimental set-up. We composed 200 ms spaced order book snapshot of BTC/USDT pair from Binance exchange. We established window size w equals to 5 ticks and depth level N equals to three and thr equals to 2.0. We expect that our classifier will have the same sign as $\frac{ma_t^{t+w} - mp^t}{mp^t}$ where ma_{t1}^{t2} is a moving average for a window of $t1 \ldots t2$ and mp^t is a mid-price at tick t. As a result, we analysed 360 thousand such windows across which 985 times was registered a non-null trend and accuracy in those cases totaled 55.74 %. Since expected percentage according to the stationary OU model is 50 %, we consider such result to be significant.

Thus, as a result, we have considered usage of this indicator as a part of our features set for RL model.

Static algorithms

For purposes to evaluate ability of RL methods to influence decision-making of static algorithms, we selected several known methods. First, we chose one of the oldest and classic market making algorithms — Non stationary OU — based mean reverting algorithm. As stated at Eq. (3), there are four parameters required for evaluation: β — trend, γ — long-term portfolio value, α — mean-reversion strength and σ — volatility. We note that according to our experiments, on short-term windows, β most of the times indistinct from zero and thus do not influence decision process. Thus, de-facto we were

using stationary OU process as denoted at Eq. (2). As for trading strategy itself, we will be utilising known distribution of OU process as denoted above. Thus, the quoting price will be as follows:

$$q = \begin{cases} \left(s,\ \gamma + K^{out}\zeta\right), & \text{if } s - \gamma > K^{in}\zeta, \\ \left(\gamma - K^{out}\zeta,\ s\right), & \text{if } s - \gamma < -K^{in}\zeta, \\ (\infty,\ -\infty), & \text{otherwise}, \end{cases} \tag{14}$$

where q is a two-length tuple of ask and bid quotes, K^{in} and K^{out} are constants, s resembles a midprice and $\zeta = \frac{\sigma}{\sqrt{2\alpha}}$ is a standard deviation in accordance with the OU model. Note that with $q \to \infty$ for ask quote or $q \to -\infty$ for bid quote will almost never match with other orders placed on exchange.

Another model we will be trying to optimise is the AS [Avellaneda, Stoikov, 2008] model. As discussed earlier, we will keep track of the reservation price $r(s, t)$ and optimal spread δ. A new order will be issued based with the following probability:

$$p\left(\delta^s\right) = Ae^{-k\delta^s}, \tag{15}$$

where A is a constant relevant to trading frequency of a specific asset, k is a constant and δ^s is an absolute difference between a quoting price (see Eqs. (5) and (6)) and current midprice. Our RL model will be trying to optimise the k, A and γ parameters in Eq. (5).

Finally, we will be optimising the "grid" strategy defined in the previous sections. As we noted, this model have a "Regression Network" which is responsible for close price prediction. We will be trying to utilize an RL model in order to predict deviation of a future close price from current midprice. Next, we will be approximating first and second derivatives of a price trend as follows:

$$\eta_1 = \frac{\partial s(t+1)}{\partial t} = s_{pred}(t+1) - s(t), \tag{16}$$

$$\eta_2 = \frac{\partial s(t+1)^2}{\partial t} = s_{pred}(t+1) + s(t-1) - 2s(1), \tag{17}$$

where $s(t)$ is closing midprice at time tick t and $s_{pred}(t)$ is a predicted price at time t. On each time tick we will classify a trend in accordance with the following equation:

$$\alpha = \begin{cases} 1, & \text{if } \eta_1 > \delta \text{ and } \eta_2 > 0, \\ -1, & \text{if } \eta_1 < -\delta \text{ and } \eta_2 < 0, \\ 0, & \text{otherwise}, \end{cases} \tag{18}$$

where δ is a constant. After a trend is defined, we will issue quotes which aiming at a fixed profit y which is also a constant. With our RL model we will also be optimising the δ and y hyper-parameters.

Technical indicators

In order to improve horizon of available data to RL model, we will tune our observable state space. In our model we will be considering the following features:

- **[bid-ask]_px_X** — price of a X level of ask and bid side,
- **[bid-ask]_qt_X** — volume of a X level of ask and bid side,
- **midprice** — average of **ask_px_1** and **bid_px_1**,
- **VWAP_X** — Volume Weighted Average Price for X levels [Amir, Masoud, 2020],

- **mean_W** — mean midprice for the last window of length W,
- **vol_W** — standard deviation for the last window of length W,
- **sigma_W** — volatility in accordance with Eq. (4),
- α**_W** — mean reversion strength in accordance with Eq. 4,
- **imbalance_X** — imbalance of the first X levels in accordance with Eq. (7),
- **microprice** — the order book microprice [Cartea, Jaimungal, Penalva, 2015] in accordance with Eq. (19)

$$\text{microprice} = \frac{\text{ask\_qt\_1} \cdot \text{ask\_px\_1} + \text{bid\_qt\_1} \cdot \text{bid\_px\_1}}{\text{ask\_qt\_1} \cdot \text{bid\_qt\_1}}, \tag{19}$$

- **open_W', max_W', min_W'** — candle stick data for a window $W' < W$,
- **jumps_W"** — number of "jumps" [Lee, Mykland, 2007] occurred within a previous window W'' detected as follows:

$$J = \begin{cases} 1, & \text{if } \dfrac{\log(mp_i) - \log(mp_{i-1})}{\phi} > \eta, \\ 0, & \text{otherwise,} \end{cases} \tag{20}$$

  where η is a constant which purposed to be as $\eta = 4.60001$ and ϕ is a variance of midprice in log-space,

- **qt_trend_[ask-bid]_W'** — volume trend as defined in Eq. (13) across window W' for ask and bid.

According to our experiments, an optimal values of X, W, W' and W'' are 3, 8192 (approx. 15 minutes), 1500 (5 minutes) and 5, respectively.

Reward considerations

As can be deduced from the Eq. (10) an RL model requires properly established reward. Since a purpose of any trading system is to generate profit, PnL is essential for reward computation.

Another important feature of any trading system is risk management which primarily linked to amount of stock at a point in time as implemented in AS and Grid models. We may leave this consideration to the model itself and make it optimise it's weights to account current position. However, it could be more efficient to include this information into reward function in order to regulate applicable risk level by the end user.

Finally, during experiments, we noted that RL models tends to enter to a "caution mode" when actions euclidean norm becomes high and thus not trades can be made. We account this phenomena by experience highly negative return streak by the model which makes it hold the quotes to prevent negative rewards. In order to counter this, we included Euclidean norm of outputted actions into reward calculations.

With that in mind, we derived the following reward function:

$$R(s, a) = r^p - c_1 q - c_2 \|a\|, \tag{21}$$

where r^p is a yielded return, $q(s)$ is current stock, c_1, c_2 are constants. Manipulation of c_1 and c_2 allows to regulate risk appetite of a model.

Experiments and evaluation

The mission of the following experiments is to determine whether RL optimization of static trading algorithms can be effective. In order to complete it, we will deploy static algorithms with RL support. As a baseline we utilized the underlying static algorithms with calibrated parameters without any RL optimisation. Note that we did not launched grid algorithm since it requires an ML-based close price predictor. Parameters for static algorithms were calibrated using a grid search approach on evaluation dataset. And finally, we will also look into direct RL market making algorithms, e. i. RL agents actions of which represent ask/bid quotes (note, that negative or extremely positive values of quotes will not yield in complete orders). Thus we will the three classes of algorithms.

We will employ three RL algorithms in this experiment — DDQN, DDPG and TD3. Each algorithm had 2000 episodes with 100 steps each for train algorithm and a single pass through test dataset with optimisations on each 100 steps. Full parameters of RL and static models are presented in Table 1.

Table 1. Model's parameters

Params	DDQN	DDPG	TD3	PPO
Q-network layers	1024 | 256 | 128	NA	NA	NA
Actor / Policy layers	NA	256 | 16	256 | 16	512 | 64
Critic / Value layers	NA	512 | 128 | 16	512 | 16	512 | 128 | 16
γ	0.8	0.9	0.9	0.99
c_1	0.7	0.6	0.6	0.6
c_2	0.2	0.2	0.3	0.3
lr	1e – 6	1e – 7	1e – 7	1e – 8
lr^y	0.99	0.993	0.993	0.995
θ^{clip}	1e – 7	1e – 8	1e – 8	1e – 8
β	0.7	0.7	0.7	NA
π^{clip}	NA	NA	NA	0.01
L^{ent}	NA	NA	NA	0.7
L^{val}	NA	NA	NA	0.8
OU^{Kin}	3.0	3.0	3.0	3.0
OU^{Kout}	1.5	1.5	1.5	1.5
AS^{α}	1.2	1.2	1.2	1.2
AS^{k}	1.2	1.2	1.2	1.2
AS^{γ}	0.7	0.7	0.7	0.7
GR^{δ}	3e – 4	3e – 4	3e – 4	3e – 4
GR^{pnl}	5e – 4	5e – 4	5e – 4	5e – 4

Results on RL market making algorithms

We run the three RL algorithms which directly outputted quotes for order composition. For discrete DDQN algorithms action space was presented as number of basis points from current mid-price. Having a set of allowable distances — [5, 30, 200] we composed all possible pairs (ask and bid quotes) totalling 9 possible actions. To that we have added another neutral action. As for continuous action spaces, those will be directly translated to relative step from a midprice. Evaluation of those models yielded PnL as presented on Fig. 2.

As a result, the DDQN, DDPG, TD3 and PPO models achieved Sharpe coefficients of 2.8, 3.1, 2.4 and 0.2, respectively. So, despite the fact that DDQN achieved considerably higher PnL than the other models, standard deviation of PnL indicates high volatility of the model output. Thus, the most

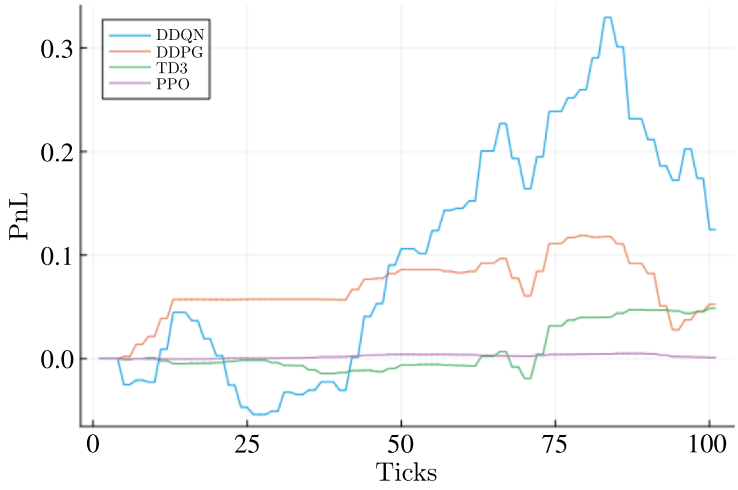

Figure 2. PnL overview of combined RL and Static model for trading. Each line corresponds with a RL model employed as noted on the figure. All models with exception of PPO ended up with distinctively positive PnL on BTCUSDT perpetual futures for the period of April 2024

stable and profitable solution in this case can be considered DDPG which achieved the highest Sharpe ratio.

We note that behaviour of agents proved to be to some degree unstable having the following loss dynamics (see Fig. 3).

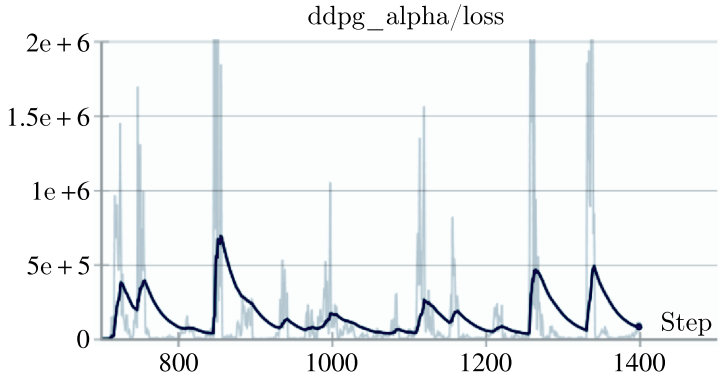

Figure 3. Loss dynamics of the direct DDPG model. Transparent blue line corresponds to factual loss trends. Blue line depicts smoothed loss for the total training period

The graph demonstrates sudden explosion of the critic loss across optimisations on evaluation data. That indicates that any market changes may have significant effect on RL model decision making.

Results on RL optimisation of OU model

In order to optimise the OU model proposed in “Static algorithms”, we defined actions of RL models as coefficients to the OU parameters, namely K^{in}, K^{out} and α as defined in Eqs. (2) and (14). Resulted PnL dynamics demonstrated on Fig. 4.

We may note that all models ended up with zero or negative PnL. Negative PnL of original OU model could be expected since modern methods utilize more sophisticated tools [Qu, Dassios, Zhao, 2021; Wang, Song, Wang, 2015]. In this case low profitability primarily linked to weak α values observed during the evaluation period. However, the RL models seems to detect low profitability of the model and corrected coefficients in such a way that make orders do not get filled. As a result, all

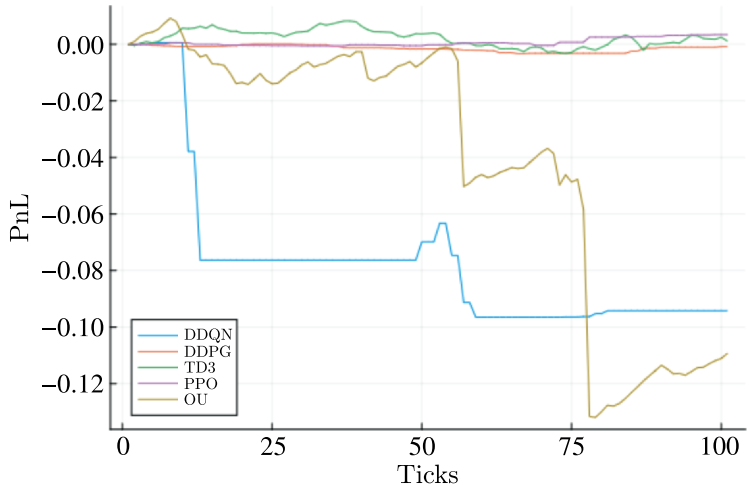

Figure 4. PnL dynamic of OU models. Each line corresponds with a RL model employed as noted on the figure. All models with exception of DDPG, TD3 and PPO ended up with distinctively negative PnL on BTCUSDT perpetual futures for the period of April 2024

models managed to improve results — DDPG, TD3 and PPO managed to escape fulfillment of orders and DDQN managed to minimize losses.

Results on RL optimisation of AS model

In order to optimise AS model, we configured RL models to optimise the k, A and γ parameters in Eq. (5). The resulted outcome of experiments presented on Fig. 5

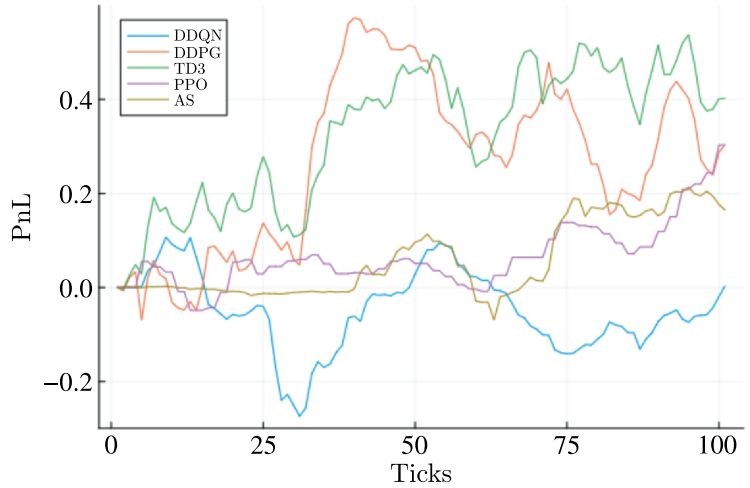

Figure 5. PnL dynamic of AS models. Each line corresponds with a RL model employed as noted on the figure. All models with exception of DDQN ended up with distinctively positive PnL on BTCUSDT perpetual futures for the period of April 2024

Experiments demonstrated promising results with DDQN, DDPG, TD3 and PPO having −0.5, 4.24, 3.5 and 2.9 Sharpe coefficients respectively. The original AS model ended up with 1.9 Sharpe coefficient.

Thus, every model with exception of DDQN was able to improve performance of raw AS model. However, similar to the direct approach, models experience significant jumps in loss acquired.

In addition, all models with exception of PPO has demonstrated significant drawdowns, representing higher risk strategy.

Results on RL optimisation of Grid model

We composed experiments on optimisation of the Grid model. As we stated earlier, our RL models were aiming at optimising the close price as well as target profitability and tolerance coefficient. The results of the experiments presented on Fig. 6. Note that we did not compose any baseline model since by design, the Grid model should have a Regression block which include a machine learning algorithm. Since that part is effectively is completed by an RL neural nets, baseline model will be similar to the ones used in experiments.

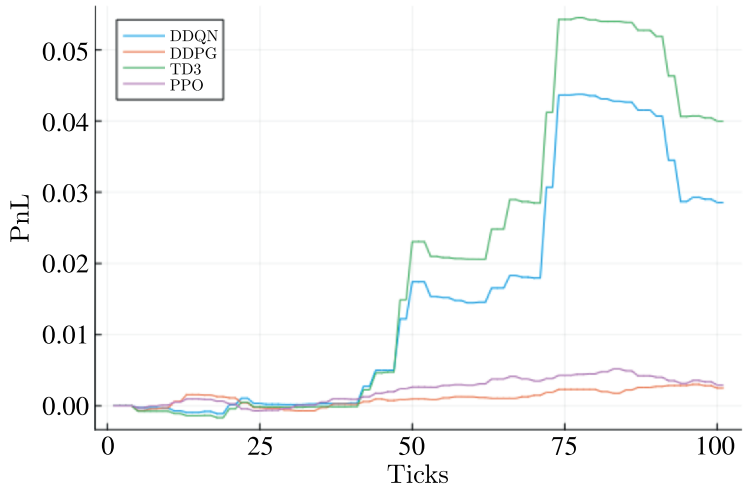

Figure 6. PnL dynamic of Grid models. Each line corresponds with a RL model employed as noted on the figure. All models with exception of DDPG and PPO ended up with distinctively positive PnL on BTCUSDT perpetual futures for the period of April 2024

As showed on the graph, DDQN and TD3 demonstrated the most promising results meanwhile PPO and DDPG ended up with zero PnL. Although PnL figures ended up less than in AS or direct approach, DDQN and TD3 have little drawdown and demonstrate comparatively stable results. We may also note that DDPG model ended up with no-action strategy meanwhile TD3, an algorithm with similar architecture ended up with profitable run. This indicates that under other random variables, DDPG could achieve similar results as TD3 and vice versa. Thus, here we might observe a phenomena of obstructed reproducibility as noted by some authors [Chaouki et al., 2020; Henderson et al., 2018].

Analysis and observations

We have evaluated 12 RL algorithms and two RL-free baselines. Summary of the results are outlined in the Table 2.

Table 2. Summary on PnL/Sharpe of experiments

Model	OU	AS	Grid	Direct
DDQN	−.09/ − 1.8	−.01/ − 0.5	.03/1.6	.11/2.8
DDPG	.00/0.1	.23/4.2	.00/0.3	.03/3.1
TD3	.00/0.2	.40/3.5	.04/1.8	.03/2.4
PPO	.00/0.3	.23/2.9	.00/0.4	.00/0.2
No RL	−.11/ − 2.9	.18/1.9	NA	NA

We may observe that Sharpe ratio for AS RL algorithms turned out to be higher the other ones. However, high profitability of such models is influenced by high profitability of non-modified AS model. Thus, if a market regime change, we may observe different situation. Similar to OU — as of selected period there were limited opportunities for the strategy execution since observed α was low. For a different period or instrument this situation could change.

Overall, our experiments showed that RL steadily improve performance of underlying static models. In addition, RL models demonstrated capability to act directly as market-making model.

However, a loss dynamic suggest that RL algorithm might have difficulties in adoption of new market realities. Specifically, as showed on Fig. 3, loss is unstable and thus it may be unreliable on greater periods of time. This could happen due to stochastic nature of price trends and constant regime switches on various time frequencies. Such behaviour could be explained by frequent adaptation of an RL model to new market regimes.

We may also note that TD3 model provides more strong results compared to other models. However, PPO rends to demonstrate more stable results amid having significantly lower PnL and Sharpe ratio.

In addition, we evaluated the models on opportunity to into high-frequency environment. We ran each model for 100 episodes consisting of 100 steps totalling 10 000 steps and 100 optimisation routines. We utilised a Julia language in order to conduct the experiments. As a result, average time per tick for DDQN, DDPG, TD3 and PPO with configuration as per Table 1 and hardware configuration as stated in Table 3 totaled 5.887 ms, 3.582 ms, 2.299 ms and 4.539 ms respectively which is within allowable time-frame for high-frequency trading [Cartea, Jaimungal, Penalva, 2015]. By using lower level languages (i. e. C++), time can potentially be significantly improved.

Table 3. Hardware configuration

Parameter	Description
Operating system	Ubuntu 22.04.4 LTS
Processor	12th Gen Intel(R) Core(TM) i7-12700H, 6 cores 4600 MHz, 8 cores 3500 MHz
Graphics	NVIDIA GeForce RTX 4060 8Gb
Julia version	Julia 1.10.3

Conclusion

We concluded experiments on application of RL in high frequency trading. We attempted to improve known static algorithms as well as apply RL models directly using cryptocurrency data from the Binance exchange.

As a result, we demonstrated that RL methods can be used both for static algorithms performance as well as direct output of relevant quotes. We also highlighted unstable tendencies in RL approach and vulnerability to market regimes changes. We also demonstrated that time taken for a model decision making is within allowable time frames for High-Frequency execution.

As for further work, we would suggest to research stability of RL approach in greater details. There could be classification of regime changes and consequences of those on RL-supported algorithms. In addition, various approaches can be experimented with such as ensemble of trading agents with different static algorithms under the hood. That could partially compensate instability on the models observed as well as mitigate excessive drawdown risks. In addition, the topic of regime changes and adaptation evaluation of an RL model to those changes could be a wide topic for further investigation. By understanding the nature of regime changes as well as ability of an RL approach to adapt to those could significantly increase robustness if a method. Moreover, further work could be done in terms of risk management in such models. In our models, maximum positions were bounded by the static algorithms themselves (i. e., in OU model, the position is clearly bounded by a constant and depended

on α and σ) and not took consideration for the current balance or price dynamics. Additional model-neutral risk-management agents could be introduced in order to adjust output positions. Application of additional instruments into a model such as options could greatly improve performance.

References

Amir G., Masoud E. M. Investigating the performance of an order imbalance based trading strategy in a high-frequency trading // Industrial Engineering & Management Systems. — 2020. — Vol. 19, No. 1. — P. 174–183.

Avellaneda M., Stoikov S. High-frequency trading in a limit order book // Quantitative Finance. — 2008. — Vol. 8, No. 3. — P. 217–224.

Bachelier L. Théorie de la spéculation // Annales scientifiques de l'École Normale Supérieure. — 1900. — 3e série, Vol. 17. — P. 21–86. — http://www.numdam.org/articles/10.24033/asens.476/

Beaulieu J. M., Jhwueng D.-C., Boettiger C., O'Meara B. C. Modeling stabilizing selection: expanding the Ornstein – Uhlenbeck model of adaptive evolution // Evolution. — 2012. — Vol. 66, No. 8. — P. 2369–2383. — https://academic.oup.com/evolut/article-pdf/66/8/2369/49987115/evolut2369.pdf

Butler M. A., King A. A. Phylogenetic comparative analysis: a modeling approach for adaptive evolution. — 2004. — https://doi.org/10.1086/426002

Carbonneau A. Deep hedging of long-term financial derivatives // Insurance: Mathematics and Economics. — 2021. — Vol. 99. — P. 327–340. — https://www.sciencedirect.com/science/article/pii/S0167668721000512

Carta S., Corriga A., Ferreira A., Podda A. S., Recupero D. R. A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning // Applied Intelligence. — 2021. — Vol. 51. — P. 889–905.

Cartea A., Donnelly R., Jaimungal S. Enhancing trading strategies with order book signals // Applied Mathematical Finance. — 2018. — Vol. 25, No. 1. — P. 1–35.

Cartea Á., Jaimungal S., Penalva J. Algorithmic and high-frequency trading. — Cambridge University Press, 2015. — https://books.google.es/books?id=5dMmCgAAQBAJ

Casas N. Deep deterministic policy gradient for urban traffic light control // arXiv preprint. — 2017. — arXiv:1703.09035

Chaouki A., Hardiman S., Schmidt C., Sérié E., de Lataillade J. Deep deterministic portfolio optimization // The Journal of Finance and Data Science. — 2020. — Vol. 6. — P. 16–30. — https://www.sciencedirect.com/science/article/pii/S2405918820300118

CoinMarketCap — markets rating agency. [Electronic resource]. — https://coinmarketcap.com/

Conegundes L., Pereira A. C. M. Beating the stock market with a deep reinforcement learning day trading system // 2020 International Joint Conference on Neural Networks (IJCNN). — 2020. — P. 1–8. — DOI: 10.1109/IJCNN48605.2020.9206938

Dong Y., Yu C., Ge H. D3PG: decomposed deep deterministic policy gradient for continuous control // International Conference on Distributed Artificial Intelligence. — 2020. — P. 40–54.

Falces Marin J., Díaz Pardo de Vera D., Lopez Gonzalo E. A reinforcement learning approach to improve the performance of the Avellaneda – Stoikov market-making algorithm // PloS One. — 2022. — Vol. 17, No. 12, P. e0277042.

Forbes Russia. Bitcoins 2.0. — 2014.

Fujimoto S., van Hoof H., Meger D. Addressing function approximation error in actor-critic methods // Proceedings of the 35th International Conference on Machine Learning. — 2018. — P. 1587–1596. — https://proceedings.mlr.press/v80/fujimoto18a.html

Guilbaud F., Pham H. Optimal high-frequency trading with limit and market orders // Quantitative Finance. — 2013. — Vol. 13, No. 1. — P. 79–94. — https://doi.org/10.1080/14697688.2012.708779

Hansen T. F., Pienaar J., Orzack S. H. A comparative method for studying adaptation to a randomly evolving environment // Evolution. — 2008. — Vol. 62, No. 8. — P. 1965–1977. — https://academic.oup.com/evolut/article-pdf/62/8/1965/49893718/evolut1965.pdf

Henderson P., Islam R., Bachman P., Pineau J., Precup D., Meger D. Deep reinforcement learning that matters // Proceedings of the AAAI conference on artificial intelligence. — 2018. — Vol. 32, No. 1.

Hinchin A. Asymptotic laws of probability theory. — ONTI NKTP, 1936.

Huang Y., Zhou C., Cui K., Lu X. A multi-agent reinforcement learning framework for optimizing financial trading strategies based on TimesNet // Expert Systems with Applications. — 2024. — Vol. 237. — P. 121502. — https://www.sciencedirect.com/science/article/pii/S0957417423020043

Itô K. On stochastic differential equations. — New York: American Mathematical Society, 1951. — Vol. 4.

Kabbani T., Duman E. Deep reinforcement learning approach for trading automation in the stock market // IEEE Access. — 2022. — Vol. 10. — P. 93564–93574.

Kaelbling L. P., Littman M. L., Moore A. W. Reinforcement learning: a survey. — 1996. — https://doi.org/10.1613/jair.301

Kramer T. Robust estimation of Ornstein – Uhlenbeck parameters. — The University of Wisconsin-Milwaukee, 2022.

Lee E. J., Eom K. S., Park K. S. Microstructure-based manipulation: Strategic behavior and performance of spoofing traders // Journal of Financial Markets. — 2013. — Vol. 16, No. 2. — P. 227–252.

Lee S. S., Mykland P. A. Jumps in financial markets: a new nonparametric test and jump dynamics // The Review of Financial Studies. — 2007. — Vol. 21. — No. 6. — P. 2535–2563. — https://academic.oup.com/rfs/article-pdf/21/6/2535/24453758/hhm056.pdf

Levi P. Two Documents by Paul Levi (16 March 1920–8 January 1921) // Historical Materialism. — 2017. — Vol. 25, No. 1. — P. 175–183. — https://brill.com/view/journals/hima/25/1/article-p175_6.xml

Li H., Huang J., Wang B., Fan Y. Weighted double deep Q-network based reinforcement learning for bi-objective multi-workflow scheduling in the cloud // Cluster Computing. — 2022. — Vol. 25, No. 2. — P. 751–768. — https://doi.org/10.1007/s10586-021-03454-6

Liu P., Zhang Y., Bao F., Yao X., Zhang C. Multi-type data fusion framework based on deep reinforcement learning for algorithmic trading // Applied Intelligence. — 2023a. — Vol. 53, No. 2. — P. 1683–1706.

Liu S., Wang B., Li H., Chen C., Wang Z. Continual portfolio selection in dynamic environments via incremental reinforcement learning // International Journal of Machine Learning and Cybernetics. — 2023b. — Vol. 14. — P. 269–279. — https://doi.org/10.1007/s13042-022-01639-y

Lo A. W. Hedge funds: an analytic perspective — updated edition. — 2010.

Lussange J., Lazarevich I., Bourgeois-Gironde S., Palminteri S., Gutkin B. Modelling stock markets by multi-agent reinforcement learning // Computational Economics. — 2021. — Vol. 57, No. 3. — P. 113–147. — https://doi.org/10.1007/s10614-020-10038-w

Merjin L., Averbuch L. Stochastic mean-reverting trend (SMART) model in quantitative finance. — 2024.

Mnih V., Kavukcuoglu K., Silver D., Rusu A. A., Veness J., Bellemare M. G., Graves A., Riedmiller M., Fidjeland A. K., Ostrovski G., Petersen S., Beattie C., Sadik A., Antonoglou I., King H., Kumaran D., Wierstra D., Legg S., Hassabis D. Human-level control through deep reinforcement learning // Nature. — 2015. — Vol. 518, No. 7540. — P. 529–533. — https://doi.org/10.1038/nature14236

Priola E., Zabczyk J. Densities for Ornstein – Uhlenbeck processes with jumps // Bulletin of the London Mathematical Society. — 2009. — Vol. 41, No. 1. — P. 41–50.

Qiu Y., Liu R., Lee R. S. T. The design and implementation of a deep reinforcement learning and quantum finance theory-inspired portfolio investment management system // Expert Systems with Applications. — 2024. — Vol. 238. — P. 122243. — https://www.sciencedirect.com/science/article/pii/S0957417423027458

Qu Y., Dassios A., Zhao H. Exact simulation of gamma-driven Ornstein – Uhlenbeck processes with finite and infinite activity jumps // Journal of the Operational Research Society. — 2021. — Vol. 72, No. 2. — P. 471–484.

Rieder S. Robust parameter estimation for the Ornstein – Uhlenbeck process // Statistical Methods & Applications. — 2012. — Vol. 21. — P. 411–436.

Rundo F., Trenta F., di Stallo A. L., Battiato S. Grid trading system robot (GTSbot): a novel mathematical algorithm for trading FX market // Applied Sciences. — 2019. — Vol. 9, No. 9. — Art. 1796. — https://www.mdpi.com/2076-3417/9/9/1796

Schulman J., Levine S., Abbeel P., Jordan M., Moritz P. Trust region policy optimization // Proceedings of the 32nd International Conference on Machine Learning. — 2015. — Vol. 37. — P. 1889–1897. — https://proceedings.mlr.press/v37/schulman15.html

Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O. Proximal policy optimization algorithms // arXiv preprint. — 2017. — arXiv:1707.06347

Shiller R. J. Measuring asset values for cash settlement in derivative markets: hedonic repeated measures indices and perpetual futures // Journal of Finance. — 1993. — Vol. 48, No. 3. — P. 911–931. — https://ideas.repec.org/a/bla/jfinan/v48y1993i3p911-31.html

Silver D., Lever G., Heess N., Degris T., Wierstra D., Riedmiller M. Deterministic policy gradient algorithms // Proceedings of the 31st International Conference on Machine Learning. — 2014. — Vol. 32, No. 1. — P. 387–395. — https://proceedings.mlr.press/v32/silver14.html

Singh S. P. Learning to solve Markovian decision processes. — 1993.

Singh V., Chen S.-S., Singhania M., Nanavati B., Kar A. K., Gupta A. How are reinforcement learning and deep learning algorithms used for big data based decision making in financial industries — A review and research agenda // International Journal of Information Management Data Insights. — 2022. — Vol. 2, No. 2. — P. 100094. — https://www.sciencedirect.com/science/article/pii/S2667096822000374

Sutton R. S., Barto A. G. Reinforcement learning: An introduction. — MIT press, 2018.

Uhlenbeck G. E., Ornstein L. S. On the theory of the Brownian motion // Phys. Rev. — 1930. — Vol. 36, No. 5. — P. 823–841. — https://link.aps.org/doi/10.1103/PhysRev.36.823

Van Hasselt H., Guez A., Silver D. Deep reinforcement learning with double q-learning // Proceedings of the AAAI conference on artificial intelligence. — 2016. — Vol. 30, No. 1.

Wah E., Wellman M. P. Latency arbitrage, market fragmentation, and efficiency: a two-market model // Proceedings of the fourteenth ACM conference on Electronic commerce. — 2013. — P. 855–872.

Wang S., Song S., Wang Y. Skew Ornstein – Uhlenbeck processes and their financial applications // Journal of Computational and Applied Mathematics. — 2015. — Vol. 273. — P. 363–382. — https://www.sciencedirect.com/science/article/pii/S0377042714003045

Wu D., Dong X., Shen J., Hoi S. C. Reducing estimation bias via triplet-average deep deterministic policy gradient // IEEE transactions on neural networks and learning systems. — 2020. — Vol. 31, No. 11. — P. 4933–4945.

Wu J., Wang C., Xiong L., Sun H. Quantitative trading on stock market based on deep reinforcement learning // 2019 International Joint Conference on Neural Networks (IJCNN). — 2019. — P. 1–8. — DOI: 10.1109/IJCNN.2019.8851831

Zhang H., Wang F., Wang J., Cui B. Robot grasping method optimization using improved deep deterministic policy gradient algorithm of deep reinforcement learning // Review of Scientific Instruments. — 2021. — Vol. 92, No. 2.