Pair Trading with Reinforcement Learning

Jie CHEN, Shoukang HU
Department of System Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, Hong Kong
{chenjie, skhu}@se.cuhk.edu.hk

Abstract

Pair trading is an important trading strategy in statistical arbitrage due to its simplicity. However, most of the trading strategies are found manually by researchers. The researchers look into the financial market, detect special patterns and program them into algorithms. Reinforcement learning, being a framework of exploiting environment and making decisions together, can be used to automate this process. In this paper, we investigate how reinforcement learning can enhance the existing strategies as well as make buy /sell decisions for pair trading. Based on our analysis, β value has more influence on the average return than the the percentile of cash and z-score in the pair trading task. To the best of our knowledge, it is the first RL algorithm for pair trading. Our code is released in https://github.com/skhu101/Pair-Trading-with-Reinforcement-Learning.

Index Terms - pair trading, automated trading, reinforcement learning

1 Introduction

With the development of information technology, more and more financial markets become electronic. Consequently, algorithmic trading and trading strategies are developing rapidly. Among all kinds of strategies, pair trading is the simplest and most popular one. In particular, many patterns of stock prices, for example, momentum and mean reverting, have been found and applied in pair trading. Currently these patterns are mainly detected by researchers based on the statistical analysis. However, the financial market is evolving quickly and old patterns become invalid rapidly. As a result, researchers need to explore the financial market and find new patterns and strategies all the time, otherwise they may not be able to make profits.

A natural idea would be automating the process of exploration of financial market and decision making. Reinforcement learning, i.e. learning what to do, how to map situations to actions, so as to maximize a numerical reward signal, fits well into this task. Specifically, the price prediction and the subsequent decision-making of sell and buy are integrated in one step and jointly optimized in line with the objective of investors. Even better, different constraints such as no short-selling, risk control schemes can also be incorporated together. The best of all is that the 'trading agent' learns by interacting directly with the financial markets and thus it can adapt itself with the evolution of markets automatically. In fact, many researchers have already used RL in different financial applications, such as enhancing existing strategies [1, 2], optimizing order execution [3–5], hedging basis risk [6] and expanding the number of involved assets [7–9].

In this paper, we shall investigate the power of reinforcement learning in pair trading. As far as we know, we are the first one to apply RL to pair trading. In particular, we shall try to enhance existing strategies, detect different patterns, such as momentum, mean-reverting and co-movement of two stocks with RL.

2 Related Work

Reinforcement learning is the framework that the system or agent explores the financial market in an iterative manner that allows them to learn the optimal trading strategy through new information. Recently, reinforcement learning methods [10, 11], built upon several fundamental theories, i.e., dynamic programming and control theory, were developed to manage the stock, bond and foreign exchange market etc. These methods can be classified into three categories: critic-only approach, actor-only approach and actor-critic approach.

The critic-only approach is the most frequently used RL method in financial markets. The idea of this approach is to learn a value function based on the expected outcome of different actions. During the decision process, the action with the best outcome is selected. The application of critic-only reinforcement learning has been first introduced by modelling the management of the portfolio as a Markov decision process [12]. [13] compares the performance of different approximation methods of the value function and show that Q-learning works better than kernel based methods. Recently, [14] extends deep Q-learning techniques to approximate the value function in the trading system. Other related critic-only approaches are used in the trading system [15–17], high frequency application [11, 18] and optimal execution [3].

The actor-only approach is the second most used approach. Instead of computing the outcome of all actions as in the critic-only approach, the actor-only approach directly learns a mapping from state to action. Thus, the design of mapping function is the key point of the actor-only approach. Early works form the mapping function with utility functions, i.e., an additive profit utility function [19] and a 'differential Sharpe ratio' utility function [20]. Inspired by the modelling power of neural networks, [21] conducts a comprehensive empirical analysis of the neural network decision function. Built upon these work, [22] further applies a three-layer neural network as the trading system by involving the risk preference and market conditions. In recent years, there is a tendency to first learn a low-dimension feature representation of the state of the environment by using deep neural network (DNN) and then to fed the learned feature into the recurrent neural network (RNN) for decision making [8, 23].

The remaining one is the actor-critic approach, which combines the advantage of both critic-only approach and actor-only approach. The key idea is that an actor chooses an action based on the state information and then a critic gives the judgement of the selected action. [24] implement both the actor and critic with neural networks and the reported performance is better than the actor-only model and supervised model performance. Other attempt is to design the actor-critic agent with fuzzy programming techniques [25].

Our work is closely related with the actor-critic approach. An actor is used to learn how to make trading decision based on the stock price patterns and then a critic judges how good the trading decision is. This differs from the traditional strategies, which is mainly done by experts with the statistical algorithms [26]. Inspired from previous RL application in financial markets, we shall try to enhance existing pair-trading strategies using RL.

3 Preliminaries

In this section, we shall introduce the fundamental idea and a basic trading strategy in pair trading. The idea of pair trading is rather simple: find a pair of stocks that exhibit similar historical price behaviour. When the price difference(See equation (2) for definition) of the two stocks is too high, then the price difference will decrease with high probability and we can exploiting this prediction and make money from it. Similarly for the case where the price difference is too low. A complete pair trading normally consists of following parts:

- Find two stocks with similar price dynamics;
- Recognize patterns (also called signal) in price dynamics and make prediction of future prices;
- Use the prediction to make trade decisions.

The first and the second tasks involve pattern recognition and the third involves decision-making w.r.t certain objective. Since deep neural network is good at learning features and reinforcement learning

can learn to make decisions by interacting directly with environment, we believe by combining existing strategies with deep reinforcement learning, better trading strategies can be obtained.

3.1 Baseline method

We shall introduce a basic method for pair trading and explain how we can make profits from it. Suppose we have two stocks and denote their prices and price changes as

$$S_1, S_2, dS_1, dS_2.$$

Assume perfect correlation between price changes of the two stocks, i.e.

$$\rho(dS_1, dS_2) = 1 \text{ or } dS_2 = \beta \cdot dS_1 + b. \tag{1}$$

Rewriting above equation gives:

$$S_{diff} := \beta dS_1 - dS_2 = -b \tag{2}$$

Thus if S_{diff} deviates from $-b$, we know that prices S_1, S_2 is mis-priced and we can make a profit by exploiting it. In reality, we don't have perfect correlation and S_{diff} is not a constant. But it does move around $-b$. With normality and i.i.d. assumptions, we can obtain that ($S = S_{diff}$)

$$\begin{aligned} &P(S > -b + \sigma) \approx 0.16, P(S < -b + \sigma) \approx 0.84, \\ &P(S \text{ go up} | S = -b + \sigma) : P(S \text{ go down} | S = -b + \sigma) \approx 1 : 5. \end{aligned} \tag{3}$$

Similarly we have

$$P(S \text{ go down} | S = -b - \sigma) : P(S \text{ go up} | S = -b - \sigma) = 1 : 5.$$

With such predictions, we can make a bet about the trend of price difference S_{diff}. We emphasize that it doesn't matter how each stock price moves, we actually make money from the movement of the price difference S_{diff}(co-movement of stock prices S_1, S_2). The specific procedure of the baseline method is detailed as below:

1. Pick a pair of stocks that show strong correlation between price changes;
2. Analyze and estimate the parameters involved (use historical data)
   (a) Do linear regression between dS_1, dS_2 to compute β; This parameter is very important since it describes the relationship of the price changes of two stocks.
   (b) Compute the mean μ and standard deviation σ of the price difference(equation (2));
3. Trade (use upcoming data)
   - Get current prices S_1, S_2 and compute dS_1, dS_2, S_{diff} and z-score

     $$Z = \frac{S_{diff} - \mu}{\sigma}$$

   - z-score measures the degree of deviation of S_{diff} from its mean, and we make trades based on it:
     - $Z > 1$, the price difference is too high, short sell stock 1 and buy stock 2 [1]
     - $Z < -1$, the price difference is too low, buy stock 1 and short sell stock 2
     - $-0.6 < Z < 0.6$, the price difference is neutral and we don't know the direction of movement, thus we close our position.
   - One more thing to determine is the size of each bet. For baseline method, we simply use 60% of the cash in hand for opening the position and the number of each stock to buy or sell is calculated in the following way:
     - Calculate the money to be used in the next bet :

       $$\text{money to use} = \text{ money in hand } * \text{percentile of cash to use}$$

[1] short sell means borrowing stocks from other investors and selling them. In the future, investors will buy the stocks back and return them to original investors. Contrary to buying stocks, short selling makes money when the underlying stock price decrease. For more, please refer to https://www.investopedia.com/terms/s/shortselling.asp

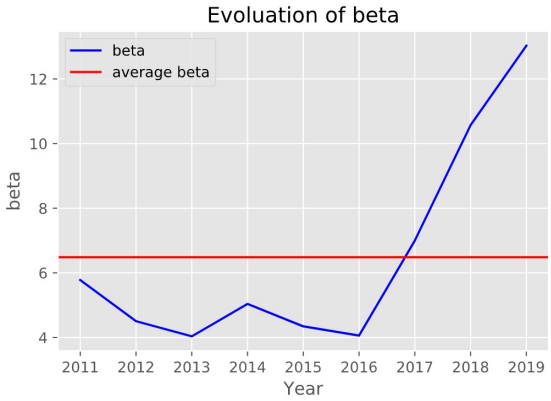


Figure 1: β estimated using each year's data

- Calculate the number of the stock which we determine to buy in last step:

$$\text{number of stock to buy} = \text{money to use} \,/\text{current price of stock to buy}$$

- Recall the definition of S_{diff}(equation (2)), thus for each bet, the number of stock 1 is always β times the number of stock 2. Given this relation, we can calculate the number of the stock to sell.

We emphasize that the percentile of cash, β value and z-score to use in each bet are the key elements in our method. Precise estimation would result in good performance of our method. So far it is quite convincing that we can make a lot of money by simply running the above method. In reality, it may not behave like that. The reasons can be summarized as below:

- The price difference(S_{diff}) does not follow the normal distribution and auto-correlation exists. As a result, systematic estimation error exists.
- The market is changing all the time. The parameters we get from historical data may not persist out of sample, which can be verified from the β estimated using each year's data in Fig. 1.
- We set simple threshold for making a bet. In reality, simply one standard deviation and hard threshold may not be good. For example, the price change move around the threshold and our algorithm may keep on opening and closing position, which causes a lot of transaction fees.
- Balance of risk and profit. Large bets will result in large profits but also high risks. If the risks are not well managed, we can go bankrupt before making a lot of money.
- Constraints such as liquidity and transaction costs. For example, during financial crisis, due to liquidity dry, proposed trade may not be successfully executed and large loss may occur.

4 RL Framework

In this section, we describe the key elements in the RL framework, like state, action space and reward function. Without loss of generality, the total time period is divided into discrete time intervals, say $t_0, t_1, \cdots, t_n = T$ and actions will be only taken at these time intervals. Statistical arbitrage is the purely technical analysis. i.e. making trading decisions purely based on market information and thus in this work, the state only consist of the current prices, current positions of selected stocks and remained cash. The details are summarized below:

- input: $(S_1, \cdots, S_N, h_1, \cdots, h_N, C)$, where N is the number of stocks we choose; S_i is the current price of i-th stock; h_i the number of I-th stock we have and it means short-selling if $h_i < 0$; C is the cash in hand.
- action: the percentile of cash, β value and z-score.

- reward: return of our portfolio $r = V^t/V^0 - 1$, where V^t denotes the portfolio value at time t.

Different limits on the trade would also be set, for example

- No short-selling, which means $p_i, k_i \geq 0$ and at each step $k_i \leq p_i$;
- No lending cash, which means $C \geq 0$ and If no enough cash for all the trade, we will do all the trade proportionally such that no cash is reserved. That is : if $\sum_{i=1}^{N} p_i k_i > C$, we will set
$$\hat{k}_i = \frac{C}{\sum_{i=1}^{N} k_i p_i} k_i$$
- Limit for position of single stock, which means we cannot put all money into one stock to avoid Black swan incidents(risk control).

5 Experiments

This section presents the experiments conducted in pair trading based on the baseline method and Proximal Policy Optimization (PPO) Algorithms [27] in Reinforcement Learning. We download ten years' daily prices of two stocks (Bank of China, China Merchants) from yahoo finance as our data set, among which the first nine years' data is used for training and the last year's data is used for evaluation. During the training, one episode consists of 200 days' data with a randomly generated starting time point. With a learning rate $5e^{-4}$ and gradient clip coefficient 5, the mini-batch (500) sampled from 15 episodes is used to update the parameters in the PPO Algorithms.

5.1 Baseline method

The average return of the baseline method with different fixed percentile of cash is shown in Table. 1. With more percentile of cash for each bet, the average return is larger. This trend is consistent with our empirical experience that the higher investment can bring the higher repayment. The highest average return (18.57%) is achieved by the baseline method with 200% percentile of cash in Table. 1. Note that in our baseline method, a fixed β value (5.33) is used.

Table 1: The average return of the baseline method with different fixed percentile of cash to use on the test data. Note that the average return is based on 100 trials.

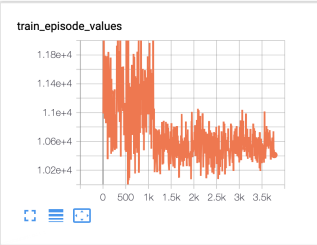
percentile of cash	0.6	1	1.5	2
average return	3.25%	6.83%	11.09%	18.57%

5.2 Learning the size of each bet

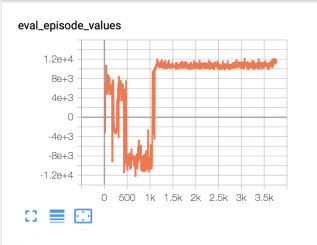
Instead of using a constant ratio 0.6 of the cash for each bet in the baseline method, we apply the PPO algorithm to learn the ratio from the candidate choices, i.e., 0.6, 1, 1.5, 2. Ratio larger than 1 denotes that we will borrow some money for investment. In the PPO algorithm, a two-layer neural network is used to predict the parameter of the categorical distribution and the current state value based on four days' state values. Combining with corresponding trading information provided by the baseline method, the cash ratio of each bet sampled from the categorical distribution is applied to make the trading decisions. While the baseline method with fixed size of bets can achieve less than 18.57% return, the RL reinforced method with $\gamma = 0.6$ achieves 20% average return as in Fig. 2. However, the RL reinforced method with $\gamma = 1.0$ does not bring a larger average return in comparison of that with $\gamma = 0.6$ in Fig. 2, which is still under further investigation.

5.3 Learning the relationship of two stock's price differences

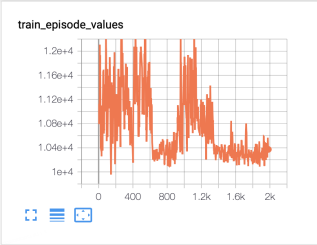
In our baseline method, we compute a β value to fit the relationship of two stocks' price differences in Eqn. 1. From our obervations and analysis in Fig. 1, such a constant β value can not describe the relationship in Eqn. 1 as the β value changes at different time intervals. In this section, the RL reinforced method is applied to learn both the size of each bet (candidates: 0.6,1,1.5,2) and β value



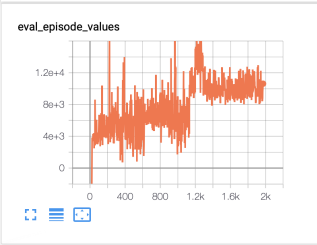
(a) Portfolio value during training (γ:0.6)



(b) Portfolio value during evaluation (γ:0.6)



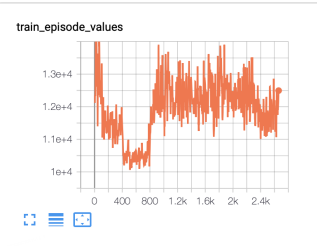
(c) Portfolio value during training (γ:1.0)



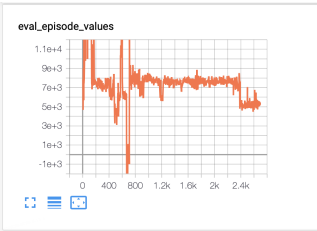
(d) Portfolio value during evaluation (γ:1.0)

Figure 2: Portfolio value for learning the size of each bet based on 4 days' state information. The average return of RL reinforced method (γ: 0.6) is about 20%.

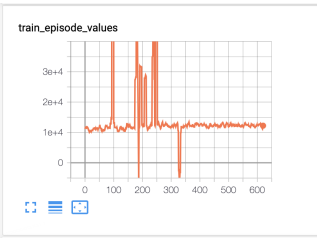
at different time intervals (candidates: 5,7,9,11,13,15). The two-layer neural network in the PPO algorithms is used to predict the parameters of the categorical distributions (the size of bet and β coefficient) and the current state value based on four days' state values. Experiments in Fig. 3 show that the average return of the RL reinforced method ($\gamma = 1.0$) is about 50% while the baseline method can only achieve at most 18.57% return. In our experimental setting, the RL reinforced method with $\gamma = 1.0$ outperforms that with $\gamma = 0.6$.



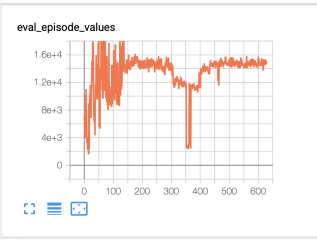
(a) Portfolio value during training (γ:0.6)



(b) Portfolio value during training (γ:0.6)



(c) Portfolio value during evaluation (γ:1.0)

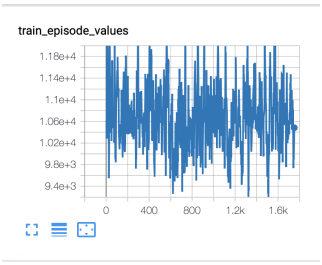


(d) Portfolio value during evaluation (γ:1.0)

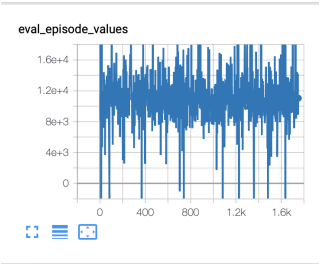
Figure 3: Porfolio value for learning the relation of price changes based on 4 days' state information. The average return of RL reinforced method (gamma: 1.0) is about 50%.

5.4 Learning the z-score of two stocks's price differences

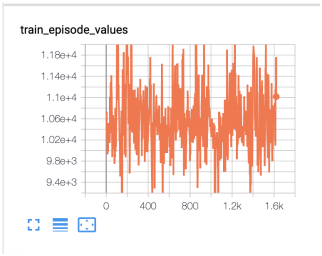
In this section, we further apply the RL reinforced method to learn the z-score, which determines the trading flags. In our experiments, we find that learning both β coefficient and z-score leads to quite large fluctuations of the portfolio value. So we fix the β coefficient and only learn the size of bet (candidates: 0.6,1,1.5,2) and z-score (candidates: -1.5,0,1.5). As shown in Fig. 4, only 18% return is achieved in the the RL reinforced method with $\gamma = 0.6, 1.0$, which may be explained as using improper β values at different time intervals.



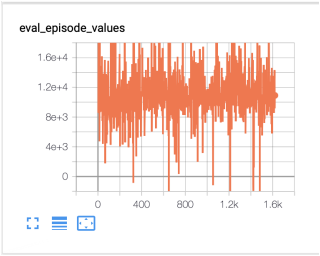
(a) Portfolio value during training (γ:0.6)



(b) Portfolio value during training (γ:0.6)



(c) Portfolio value during evaluation (γ:1.0)

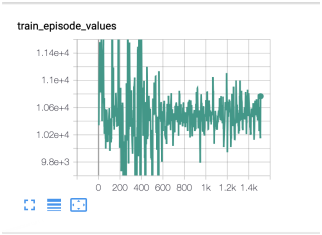


(d) Portfolio value during evaluation (γ:1.0)

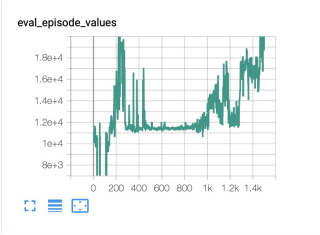
Figure 4: Portfolio value for learning the z-score of price changes based on 4 days' state information. The average return of RL reinforced method (gamma: 0.6, 1.0) is about 18%.

5.5 The effects of predicting actions based on different length of price information

In this section, we further investigate the effects of long term stock price information on the trading decisions. By predicting the action based on 10 day's state information, about 70% average return can be obtained, which outperforms that based on 4 day's state information. The result verifies that stock price in recent time intervals can help make good trade decisions.



(a) Portfolio value during training (γ:1.0)



(b) Portfolio value during training (γ:1.0)

Figure 5: Portfolio value for learning the relation of price changes based on 10 days' state information. The average return of RL reinforced method (gamma: 1.0) is about 70% based on 10 days' state information.

6 Discussion

In this work, we investigate the power of reinforcement learning in pair trading. Despited the performance achieved, there does exist some problems and limitations in our experiments. The first one is overfitting. Ten years' price data of two stocks is obviously too limited and learned features may not be universal, which reduce the generalization ability of our algorithm. One possible solution is to follow the suggestions by Justin Sirignano and Rama Cont [28] to train a general neural network that extracts universal features from stock price data. Then the trained parameters can be used as the initialization of the neural network in our task. In addition, we only consider returns in this paper. But returns come with risk together. Although we achieve good performance on returns, we don't know whether it is because of the good prediction power of our algorithm or we actually take high risk in our trade decisions. Normally Shape ratio is a good candidate for balancing return and risk. However, it can only be calculated after the trading is over, which would be impossible to proceed in our trading decisions. We look forward to finding the new measurement that can capture both the return and risk, and we also hope the new measurement is non-sparse that can make the training easier.

Besides problems and limitations, we also plan to investigate the following questions in pair training:

- Can RL learn to select a pair of stocks?
- Can we only use RL to make trading decisions?
- Can we involve more stocks/assets at the same time?

7 Conclusion

In this work, we investigate how reinforcement learning can enhance the existing strategies as well as make buy/sell decisions for pair trading. The experiments show that the RL method can achieve higher average return than the baseline method. According to our analysis, β value has more influence on the average return than the the percentile of cash and z-score. In the future, we will give more detailed analysis in how RL interacts in the financial market.

References

[1] Z. Tan, C. Quek, and P. Y. Cheng. Stock trading with cycles: A financial application of anfis and reinforcement learning. *Expert Systems with Applications*, 38(5):4741–4755, 2011.

[2] D. Eilers, C. L. Dunis, H.-J. von Mettenheim, and M. H. Breitner. Intelligent trading of seasonal effects: A decision support algorithm based on reinforcement learning. *Decision support systems*, 64:100–108, 2014.

[3] Y. Nevmyvaka, Y. Feng, and M. Kearns. Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd international conference on Machine learning*, pages 673–680, 2006.

[4] L. Noonan. Jpmorgan develops robot to execute trades. *URL https://www. ft. com/content/16b8ffb6-7161-11e7-aca6-c6bd07df1a3c*, 2017.

[5] M. Terekhova. Jpmorgan takes ai use to the next level. *URL https://www. businessinsider. de/jpmorgan-takes-ai-use-to-the-next-level-2017-8*, 2017.

[6] S. Watts. Hedging basis risk using reinforcement learning. Technical report, Working Paper, University of Oxford, 2015.

[7] S. Kaur. Algorithmic trading using sentiment analysis and reinforcement learning.

[8] Z. Jiang, D. Xu, and J. Liang. A deep reinforcement learning framework for the financial portfolio management problem. *arXiv preprint arXiv:1706.10059*, 2017.

[9] Z. Jiang and J. Liang. Cryptocurrency portfolio management with deep reinforcement learning. In *2017 Intelligent Systems Conference (IntelliSys)*, pages 905–913. IEEE, 2017.

[10] N. Kanwar et al. *Deep Reinforcement Learning-Based Portfolio Management*. PhD thesis, 2019.

[11] J. Cumming, D. Alrajeh, and L. Dickens. An investigation into the use of reinforcement learning techniques within the algorithmic trading domain. *Imperial College London: London, UK*, 2015.

[12] R. Neuneier. Optimal asset allocation using adaptive dynamic programming. In *Advances in Neural Information Processing Systems*, pages 952–958, 1996.

[13] F. Bertoluzzo and M. Corazza. Testing different reinforcement learning configurations for financial trading: Introduction and applications. *Procedia Economics and Finance*, 3:68–77, 2012.

[14] O. Jin and H. El-Saawy. Portfolio management using reinforcement learning. *Stanford University*, 2016.

[15] M. A. Dempster, T. W. Payne, Y. Romahi, and G. W. Thompson. Computational learning techniques for intraday fx trading using popular technical indicators. *IEEE Transactions on neural networks*, 12(4):744–754, 2001.

[16] Y. Chen, S. Mabu, K. Hirasawa, and J. Hu. Trading rules on stock markets using genetic network programming with sarsa learning. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1503–1503, 2007.
[17] Y. Gu, S. Mabu, Y. Yang, J. Li, and K. Hirasawa. Trading rules on stock markets using genetic network programming-sarsa learning with plural subroutines. In *SICE Annual Conference 2011*, pages 143–148. IEEE, 2011.
[18] A. A. Sherstov and P. Stone. Three automated stock-trading agents: A comparative study. In *International Workshop on Agent-Mediated Electronic Commerce*, pages 173–187. Springer, 2004.
[19] J. Moody and M. Saffell. Learning to trade via direct reinforcement. *IEEE transactions on neural Networks*, 12(4):875–889, 2001.
[20] W. F. Sharpe. Mutual fund performance. *The Journal of business*, 39(1):119–138, 1966.
[21] C. Gold. Fx trading via recurrent reinforcement learning. In *2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings.*, pages 363–370. IEEE, 2003.
[22] M. A. Dempster and V. Leemans. An automated fx trading system using adaptive reinforcement learning. *Expert Systems with Applications*, 30(3):543–552, 2006.
[23] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3):653–664, 2016.
[24] H. Li, C. H. Dagli, and D. Enke. Short-term stock market timing prediction under reinforcement learning schemes. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 233–240. IEEE, 2007.
[25] S. D. Bekiros. Heterogeneous trading strategies with adaptive fuzzy actor–critic reinforcement learning: A behavioral approach. *Journal of Economic Dynamics and Control*, 34(6):1153–1170, 2010.
[26] A. Pole. *Statistical arbitrage: algorithmic trading insights and techniques*, volume 411. John Wiley & Sons, 2011.
[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
[28] J. Sirignano and R. Cont. Universal features of price formation in financial markets: Perspectives from deep learning (march 16, 2018). *Available at SSRN: https://ssrn.com/abstract=3141294 or http://dx.doi.org/10.2139/ssrn.3141294*.