**LANG 4030 Interim Report**

**Cover page**

**FYP/FYT Project title: Machine Learning-based Quantitative Trading Strategies**

**FYP/FYT Project code: GO1**

| **Full Names of group members** | **Student ID** | **Email address** | **LANG 4030 section** |
|---|---|---|---|
| Leung Pak Hei, Marco | 20690382 | phleungaf@connect.us | T01 |

| **Sections of the report** | **Person responsible** | **Word count** |
|---|---|---|
| Introduction | Leung Pak Hei, Marco | 202 |
| Objectives | Leung Pak Hei, Marco | 130 |
| Literature survey | Leung Pak Hei, Marco | 222 |
| Methodology | Leung Pak Hei, Marco | 481 |
| Progress | Leung Pak Hei, Marco | 542 |
| References | Leung Pak Hei, Marco | |

## **Introduction**

The stock market is a complex and dynamic system. Traditional statistical models have limitations in capturing the complex relationships between stocks and other market variables [1][2]. In contrast, with the recent advancements in deep learning, it has become possible to develop more accurate models for stock prediction. Due to the robustness of the deep learning model, this project focuses on applications of deep learning in the stock market. The project includes three major strategies: K-means-based pair trading, LSTM-based portfolio optimization, and reinforcement learning-based trading. The K-means-based pair trading strategy involves trading pairs of stocks simultaneously under the risk-neutral assumption [3]. The LSTM-based portfolio optimization strategy uses RNN to optimize the portfolio composition[4][5]. The reinforcement learning-based trading strategy involves training an agent to make trading decisions based on rewards received[6].

To evaluate the performance of these strategies, a back-testing framework is used. Additionally, an automatic trading system is developed to test the strategies in real-time trading. The performance of the strategies would also be used to compare traditional pairs trading, portfolio optimization, and the benchmark market return.

This project aims to provide a comprehensive understanding of deep learning in the stock market and its potential benefits for trading. Ultimately, the goal of this project is to provide valuable insights into the use of deep learning for trading.

## Literature Review

Traditional pair trading, portfolio optimization, and reinforcement learning model research are reviewed.

**Traditional portfolio optimization:**

Introduced by Harry Markowitz in 1952, mean-variance optimization, and risk parity optimization are one of the most popular portfolio optimization methods [7] [8][9][10]. However, these traditional approaches have some limitations, such as assuming a normal distribution of asset returns and failing to account for the future value of the assets. Recently, deep learning techniques have gained attention as a potential solution to overcome these limitations. It could be worth considered to explore the improvement of such a strategy using LSTM [11].

**Traditional Pair Trading:**

Traditional pair trading is a strategy that involves buying underpriced stock and simultaneously short-selling overpriced stock that is highly correlated based on the risk-neutral assumption [12][13]. Recently, machine learning techniques like K-means have been applied to pair trading to overcome some of the limitations of traditional methods[14]. It could be tested in pair trading to assess the performance.

**Reinforcement Learning:**

Reinforcement learning (RL) is a branch of machine learning that enables agents to learn optimal behavior through interactions with an environment. Notable RL algorithms include Q-learning, policy gradient methods, and actor-critic methods. RL has also been integrated with deep learning such as deep Q-networks and deep reinforcement learning[15]. It could be potentially applied in the stock trading application.

**The motivation for the project:**

The reviewed literature suggests that deep learning has shown the capability to solve complex problems and potentially could integrate into traditional trading and boost performance. Further research and experimentation are worth exploring.

## Objectives

The primary objective of this project is to assess the effectiveness of three deep learning-based trading strategies in the stock market: K-means-based pair trading, LSTM-based portfolio optimization, and reinforcement learning-based trading. The project seeks to accomplish the following specific objectives:

1. To back-test the performance of each of the three deep learning-based trading strategies using historical data.
2. To develop an automatic trading system that integrates the three deep learning-based trading strategies for real-time trading.
3. To evaluate the performance of the automatic trading system in a simulated real-time environment and compare it with traditional trading strategies.

The project aims to provide a comprehensive understanding of the potential of deep learning-based trading strategies. Additionally, the automatic trading system developed in this project may serve as a basis for further research and development of deep learning-based trading systems.

## Methodology

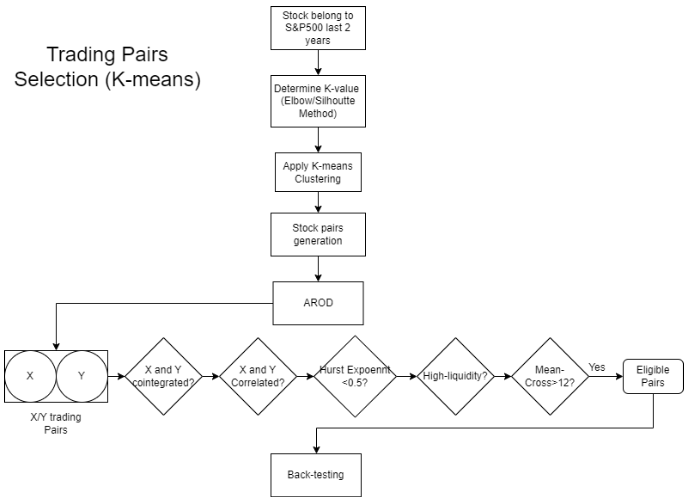
The methodology mainly consists of three parts: machine learning model research, evaluation of trading strategies, and the integration of trading systems. The data input and logic of the machine learning models and the implementation details of the trading system would be explained as follows.

### Implementation of K-means pairs trading:

The K-means clustering algorithm is applied to group stocks into clusters based on the Euclidean distance between two stocks under S&P500. The optimal number of clusters is determined by the elbow method or silhouette method [16].

Then, for each cluster, a pair of stocks is selected using Principal Component Analysis and sorted using statistical tests, such as correlation test, and cointegration test [17][18][19]. Once the stock pairs are selected, the price ratio between the two stocks in each pair is then calculated. When the absolute ratio exceeds the threshold, a long position is taken in the underpriced stock and a short position is taken in the overpriced stock, with the expectation that the ratio will converge back to its mean[20].

Finally, the performance of the strategy is backtested and compared to a benchmark index.

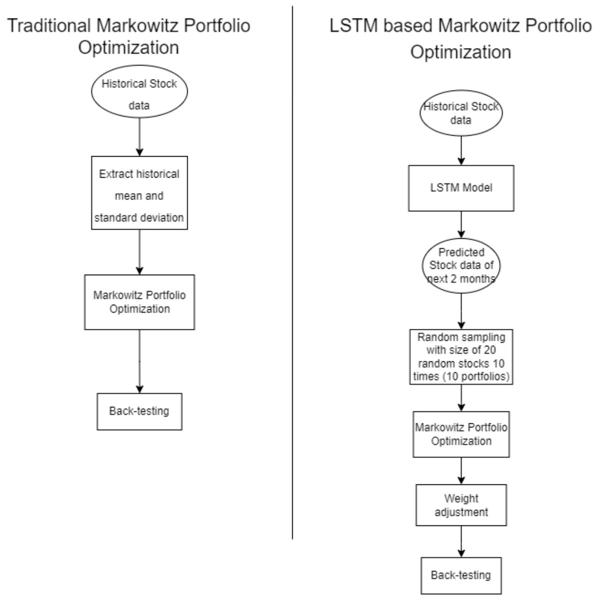


(Figure 2: Implementation details of K-means-based pair trading)

### Implementation of the LSTM Portfolio Optimization:

The LSTM-based Portfolio optimization was implemented using Keras, and TensorFlow. The closing prices of stocks from the S&P 500 will be normalized before being fed into the LSTM neural network, which consisted of LSTM layers, a dropout layer, and a dense output layer. The loss function used was mean squared error, and the optimizer used was Adam. Random sampling is done before optimization to reduce the chance of overfitting.

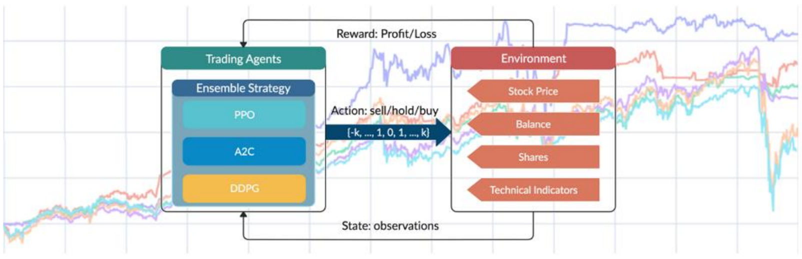
The Markowitz optimization was then done by iteratively adjusting the portfolio weights based on the covariance matrix obtained from the LSTM by quadratic programming (QP) and was solved using the CVXOPT library. The result will then be back-tested and compared to the traditional optimization.


Traditional Markowitz Portfolio Optimization
Historical Stock data
Extract historical mean and standard deviation
Markowitz Portfolio Optimization
Back-testing
LSTM based Markowitz Portfolio Optimization
Historical Stock data
LSTM Model
Predicted Stock data of next 2 months
Random sampling with size of 20 random stocks 10 times (10 portfolios)
Markowitz Portfolio Optimization
Weight adjustment
Back-testing

(Figure 3: Comparison of LSTM-based Portfolio Optimization to traditional Portfolio Optimization)

**Implementation of the Reinforcement learning-based trading:**

Our reinforcement learning (RL) trading involves using a Markov Decision Process (MDP) with states, actions, and rewards [21]. The state space comprises prices of SPX and other statistical indicators such as MA, APX, and SVI [22]. The action space consists of “buy”, “sell”, or “hold” positions. The reward function is designed to maximize the risk-adjusted return. RL algorithms such as Q-learning, Deep Q-Networks, and Policy Gradient methods are used to learn an optimal trading policy. It will then be evaluated and compared to the benchmark S&P500.



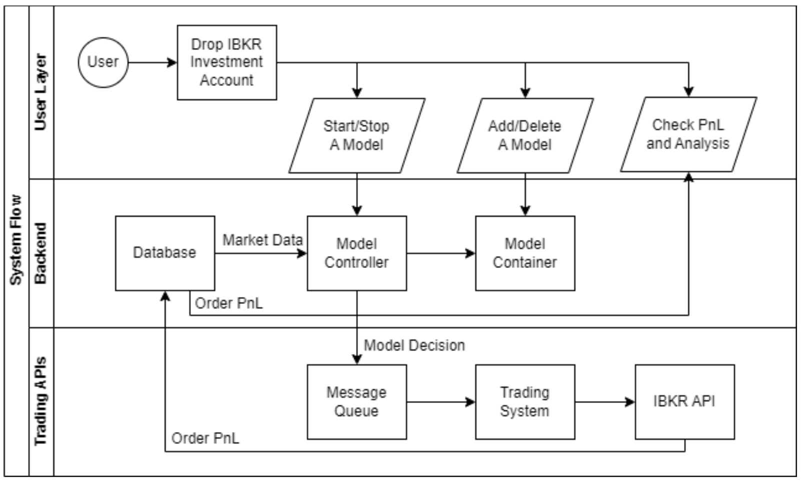
(Figure 4: Illustration of the training of reinforcement learning)

## Integration and functionality of the algometric trading system

After the machine learning models are developed, the algometric system used on Interactive Broker is integrated. The system is implemented using Python. The IBKR API is utilized for accessing real-time market data and executing trades.

A database management system (DBMS) such as MySQL or PostgreSQL is used to store historical market data, and trade execution results. The system also employs a message queue RabbitMQ to manage real-time data streams and communication between components of the system.

To ensure efficient and secure integration, the system is deployed on a cloud-based platform such as Amazon Web Services (AWS) or Microsoft Azure to ensure high availability and reliability.



(Figure 5: Swimlane diagram of the system flow)

## Work completed & Evaluation on the machine learning models

The machine learning model development and the back-testing results of the above-mentioned strategies are completed. The performance of the strategies would be evaluated and compared with the traditional strategies as follows.

**Performance of K-means Pair Trading:**

The performance of the pair trading strategy was assessed. Figure 5 shows the back-testing results of the pair trading. After sorting by the correlation test, cointegration test, and Hurst exponent test, 210 trading pairs are found. The average return of the 210 pairs is 18% over 252 trading days. The Sharpe ratio is 1.24 due to a low standard deviation (0.8). Machine learning-driven pair trading proved the capability to provide an alternative way to find pairs, compared to the traditional pairs trading results proposed by Elliott, and Hoek [23].

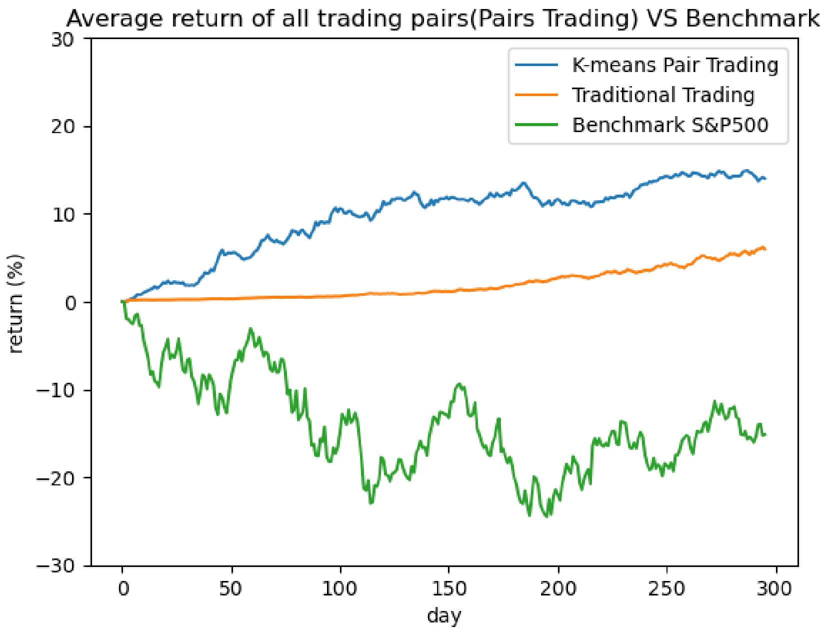


Figure 6: Performance of K-means pair trading compared to traditional pair trading and benchmark S&P500

**Performance of the LSTM Markowitz portfolio optimization strategy:**

The performance of the LSTM portfolio optimization strategy was evaluated. Table 2 illustrates examples of the weight allocation of the LSTM portfolio optimization. Based on the back-testing results, all of the portfolios have a Sharpe ratio greater than 0.8, with an average annualized return of 21% and annualized standard deviation of 0.25, which overperformed the benchmark S&P500 performance and the traditional portfolio optimization [24]. The idea of LSTM optimization proved to be a more profitable way to construct a portfolio.

Table 2: Examples of weights allocation and the performance portfolios generated by LSTM Markowitz Optimization

| Markowitz Optimization Summary (Jan 2018- Dec 2022) (10 trails of 15 random stocks from SPY) | | | |
|---|---|---|---|
| Model | Combination of tickers and weights (nearest integer) | Annualized Returns | Annualized risk |
| Sharpe ratio | [AAPL,AMZN,JNJ,NFLX,TSLA,TSM]=[14,17,14,17,14] | 0.27 | 0.253 |
| Min risk | [AMZN,JNJ,KO,NLFX,TSM,XOM]=[5,42,38,1,7,5] | 0.12 | 0.15 |
| Sharpe ratio | [JNJ,KO,TSLA,TSM,XOM,ZM]=[6,31,41,11,8,2] | 0.32 | 0.32 |
| Min risk | [JNJ, KO, TSM, XOM, ZM]=[43,35,5,7,7] | 0.12 | 0.17 |
| Sharpe ratio | [A,TSM,XOM,ZM]=[49,22,16,10] | 0.19 | 0.25 |
| Min risk | [A,BCA,F,GOOGL,SPG,TSM,XOM,Z,ZM]=[31,3,2,15,2,8,30,8] | 0.34 | 0.40 |
| Sharpe ratio | [A,NET,NVDA,OXY]=[43,32,16,8] | 0.19 | 0.25 |
| Min risk | [A,BCA,F,GOOGL,SPG,TCOM]=[43,10,6,28,5,6] | 0.15 | 0.25 |
| Sharpe ratio | [NET,NVDA,ORCL,OXY,TGT]=[26,13,14,7,39] | 0.32 | 0.36 |
| Min risk | [F,ICE,ORCL,SPG,TCOM,TGT]=[2,50,22,3,8,16] | 0.14 | 0.22 |

**Performance of reinforcement learning models :**

The performance of the reinforcement learning models was evaluated. Table 3 shows the back-testing results of the respective reinforcement learning trading strategies. Compared to the benchmark S&P500 performance, all Reinforcement learning models outperformed in terms of the Sharpe ratio, T-test, and Levene test. Among all Reinforcement learning models, the Deep Q-Leaning model with the ensemble method performed the best. Table 4 shows the back-testing results of different ensemble methods integrated with the deep Q-learning model. Among all, the ensemble method has a higher sharper ratio (1.3), lower maximum drawdown (-9.7%), and lower annualized volatility (9.7%). This is also in line with the theoretical assumption that the ensemble method can deal with randomness and short-term time-series trends.

Table 3: An Example of the reinforcement learning model (Q-Learning) to the Benchmark

| | Hold Consistently | Random Action | Rule-Based | OLS | Q-Learner |
|---|---|---|---|---|---|
| Buy % | 0% | 34% | 51% | 53% | 43% |
| Sell % | 0% | 34% | 48% | 47% | 33% |
| Hold % | 100% | 34% | 48% | 47% | 24% |
| Sharpe Ratio | +0.73 | +0.74 | +0.77 | +0.89 | +1.11 |
| Information Ratio | -0.34 | -0.32 | -0.24 | +0.07 | +0.52 |
| Mean Daily Return | +0.043% | +0.044% | +0.045% | +0.052% | +0.064% |
| Mean Return Days After Buying | N\A | +0.059% | +0.049% | +0.050% | +0.045% |
| Mean Return Days After Holding | +0.043% | +0.013% | -0.0204% | -0.123% | +0.023% |
| Mean Return Days After Selling | N\A | +0.058% | +0.046% | +0.056% | +0.117% |
| Volatility | 0.007 | 0.007 | 0.007 | 0.009 | 0.01 |
| T-test | >0.27 | >0.24 | >0.25 | >0.3 | >0.32 |
| Levene Test | <0.141 | <0.51 | <0.94 | >*0.01 | >*0.01 |

Table 4: Comparisons of different ensemble methods with Deep Q-Learning

| (2016/01/04-2020/05/08) | Ensemble (Ours) | PPO | A2C | DDPG | Min-Variance | DJIA |
|---|---|---|---|---|---|---|
| **Cumulative Return** | 70.4% | 83.0% | 60.0% | 54.8% | 31.7% | 38.6% |
| **Annual Return** | 13.0% | 15.0% | 11.4% | 10.5% | 6.5% | 7.8% |
| **Annual Volatility** | 9.7% | 13.6% | 10.4% | 12.3% | 17.8% | 20.1% |
| **Sharpe Ratio** | 1.30 | 1.10 | 1.12 | 0.87 | 0.45 | 0.47 |
| **Max Drawdown** | -9.7% | -23.7% | -10.2% | -14.8% | -34.3% | -37.1% |

## Work in Progress

The preliminary research on the machine learning models was completed. For the real-time algometric trading system, the integration and the implementations are still being developed.

For the front-end environment, an application interface written by React.js has been developed. Figure 7 also illustrates the current UI/UX design of the trading system. However, some designed interfaces, such as the real-time data reporting and login portal, and some dynamic artistic features are not yet completed.

For the back-end environment, the framework, database, and inter-process communication software have been developed by Flask, Redis, and rabbitMQ, with JSON orders. A Flask API server was also developed to allow users to get data analysis and results. However, functions like public concurrent trading, and real-time data refreshing, are still being developed.

The trading system is expected to be fully implemented in late February and will be traded on the Interactive Broker portal. By the time of the final presentation, 3-month trading data should be available for the evaluation of the performance of the real-time trading.

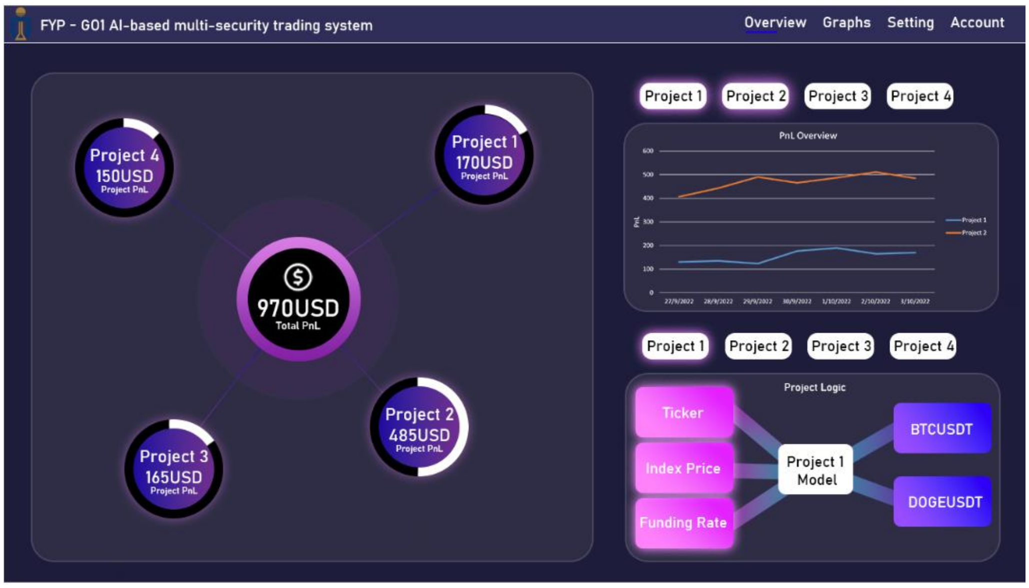


Figure 7: The current user interface of the trading system

## Overall evaluation

The progress of the project is on track and has been satisfactory. The completion of reinforcement learning trading, K-means pair trading, and LSTM portfolio optimization, along with some features of the front-end and back-end of the trading system are notable milestones. The expected completion of the fully implemented trading system is by the end of February. The project is expected to deliver a valuable tool for those involved in trading and investment.

## Reference

[1]Campbell, J. Y., & MacKinlay, A. C. (1997). The econometrics of financial markets. Princeton University Press.

[2]Lo, A. W. (2018). Adaptive markets: Financial evolution at the speed of thought. Princeton University Press.

[3] Liu, Q., Liu, S., Yang, Y., & Dai, X. (2016). A novel K-means clustering method with improved initialization for pairs trading. Neurocomputing, 214, 618-625.

[4] Kao, P. H., Hsu, Y. C., & Chen, Y. W. (2019). Forecasting volatility with long short-term memory network for optimal portfolio construction. The North American Journal of Economics and Finance, 48, 338-351.

[5] Zheng, Y., Wang, J., & Guo, H. (2020). Portfolio optimization with long short-term memory networks. Expert Systems with Applications, 157, 113471.

[6] Engelbrecht, A. P., & Cloete, I. (2004). Using reinforcement learning to trade in the stock market. Proceedings of the 2004 IEEE International Conference on Systems, Man, and Cybernetics, 4, 3733-3738.

[7] Markowitz, H. (1952). Portfolio selection. The Journal of Finance, 7(1), 77-91.

[8] Maillard, S., Roncalli, T., & Teiletche, J. (2010). The properties of equally weighted risk contribution portfolios. Journal of Portfolio Management, 36(4), 60-70.

[9] DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naïve diversification: How inefficient is the 1/N portfolio strategy?. Review of Financial Studies, 22(5), 1915-1953.

[10] Rockafellar, R. T., & Uryasev, S. (2002). Conditional value-at-risk for general loss distributions. Journal of Banking & Finance, 26(7), 1443-1471.

[11] Heaton, J., Polson, N., & Witte, J. (2017). Deep learning for finance: Deep portfolios. Applied Stochastic Models in Business and Industry, 33(1), 3-12.

[12] Gatev, E., Goetzmann, W. N., & Rouwenhorst, K. G. (2006). Pairs trading: performance of a relative-value arbitrage rule. Review of financial studies, 19(3), 797-827.

[13] Zhang, G., & Chen, X. (2016). A comprehensive study on pairs trading. Finance Research Letters, 16, 145-154.

[14] Tsantekidis, A., Passalis, N., Tefas, A., & Kanniainen, J. (2017). Using deep learning to detect price change indications in financial markets. Applied Soft Computing, 62, 995-1006.

[15] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

[16] J. C. Bezdek, R. J. Hathaway, "Some Notes on Alternatives to Spherical Clustering," IEEE Trans. Syst. Man Cybern. SMC-6(12): 869-876 (1976).

[17] Jolliffe, I. T. (2011). Principal component analysis (2nd ed.). Springer.

[18] Johansen, S. (1995). Likelihood-based inference in cointegrated vector autoregressive models. Oxford: Oxford University Press.

[19] Hurst, H. E. (1951). Long-term storage capacity of reservoirs. Transactions of the American Society of Civil Engineers, 116, 770-808.

[20] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. Addison-Wesley.

[21] R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.

[22] Kirkpatrick II, C. D., & Dahlquist, J. R. (2010). Technical analysis: The complete resource for financial market technicians. FT Press.

[23] Elliott, R. J., Hoek, J. V. D., & Malcolm, J. K. (2005). The profitability of pairs trading: a cointegration approach. Journal of Financial and Quantitative Analysis.

[24] Markowitz, H. (1952). Portfolio selection. The Journal of Finance