Adaptive rolling window selection for minimum variance portfolio estimation based on reinforcement learning

Bruno Gašperov*, Fredi Šarić, Stjepan Begušić, Zvonko Kostanjčar
* Laboratory for Financial and Risk Analytics, Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia

***Abstract* - When allocating wealth to a set of financial assets, portfolio optimization techniques are used to select optimal portfolio allocations for given investment goals. Among benchmark portfolios commonly used in modern portfolio theory, the global minimum variance portfolio is becoming increasingly popular with investors due to its relatively good performance which stems from both the low-volatility anomaly and the avoidance of the estimation of first moments i.e. mean returns. However, estimates of minimum variance portfolio weights significantly depend on the size of the rolling window used for estimation, especially considering the non-stationarity of the underlying market dynamics. In this paper, we use a model-free policy-based reinforcement learning framework in order to directly and adaptively determine the optimal size of the rolling window. Training is done on a subset of trading stocks from the NYSE. The resulting agent achieves superior performance when compared against multiple benchmarks, including those with fixed rolling window sizes.**

***Keywords* – reinforcement learning; portfolio optimization; covariance estimation**

I. INTRODUCTION

Despite the implications of the Capital Asset Pricing Model (CAPM), multiple empirical studies show that the global minimum variance (GMV) portfolio yields surprisingly high average returns [1] [2] [3]. This is related to the general stylized fact of stock markets - the low volatility anomaly – an observation that low-beta (low-volatility) stocks overperform high-beta (high-volatility) stocks [4]. Furthermore, calculation of the GMV portfolio weights does not require the estimation of first moments (expected stock returns) which is known to be notoriously difficult [5]. It is in part due to the aforementioned advantages that investors have lately been increasingly turning to the GMV portfolio [6]. (For performance of the GMV portfolio on a subset of considered stocks see Figure 1.)

However, it should be noted that estimates of minimum variance portfolio weights significantly depend on the size T of the rolling window used for estimation of the sample return covariance matrix. The optimal size of the window can be reasonably expected to depend on the presence of correlation changes in the time series of returns, with shorter optimal window sizes possibly corresponding to periods that include correlation change points (and vice versa). To the authors' knowledge, the amount of research done in this area is quite scarce with only a few relevant studies [7]. Motivated by this, our goal in this paper is to develop a method for adaptive rolling window selection based on reinforcement learning. The use of reinforcement learning is warranted due to intrinsic sequentiality and non-stationarity of the problem as well as the absence of data labels (as that the optimal choice of the rolling window size at each time step is not known).

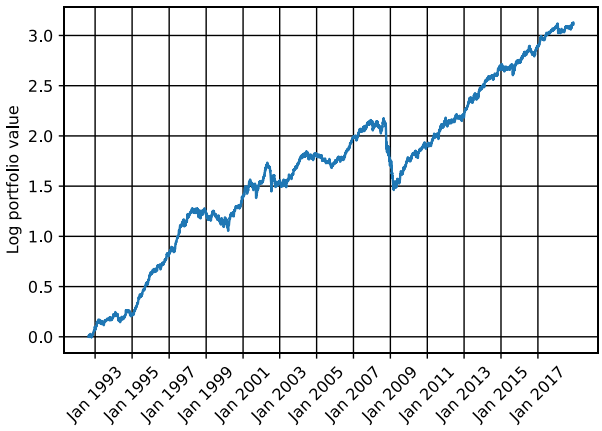

Figure 1: Global minimum variance portfolio log value for the considered 20 NYSE stocks on the whole dataset for a fixed size of the rolling estimation window (one trading year i.e. T=252)

II. PROBLEM DEFINITION

Under the minimum variance portfolio optimization strategy, historical data is firstly used to obtain the sample covariance matrix $\hat{\Sigma}$ which is given by the following expression:

$$\hat{\Sigma} = \frac{1}{T-1}\sum_{t=1}^{T}(\boldsymbol{r_t} - \bar{\boldsymbol{r}})(\boldsymbol{r_t} - \bar{\boldsymbol{r}})^{\mathrm{T}}$$

where T is the size of the estimation window, $\boldsymbol{r_t}$ return at time t and $\bar{\boldsymbol{r}}$ the sample mean of the returns.

The GMV portfolio w^* is then given as the solution to the following optimization problem:

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\hat{\boldsymbol{\Sigma}}\mathbf{w}$$

under the constraint:

$$\mathbf{1}^{\mathrm{T}}\boldsymbol{w} = 1$$

The analytical solution exists and is given by:

$$w^* = \frac{\widehat{\boldsymbol{\Sigma}}^{-1}\mathbf{1}}{\mathbf{1}^{\mathrm{T}}\widehat{\boldsymbol{\Sigma}}^{-1}\mathbf{1}}$$

Given that the only constraint is that portfolio weights should sum up to 1, short selling is allowed. Note that the sample covariance matrix depends on T, i.e. the size of the rolling window used for estimation. Considering that the sample covariance is used as an input to the optimization step, the obtained portfolio weights consequently also depend on T.

Despite the unbiasedness of the sample covariance matrix, it is corrupted by a significant degree of estimation error, especially when the number of assets N is commensurate with the size of the estimation window T. Various classes of methods have been proposed to ameliorate this problem, including factor model estimators and shrinkage estimators [8] [9] [10]. Unlike such methods, in this paper we do not attempt to directly „clean“ the covariance matrix from noise but instead use reinforcement learning as an auxiliary tool for adaptive rolling window selection. However, it is possible to combine such methods with our adaptive framework.

Under our framework, at each day (t) the trader makes the choice of the size (length) of the rolling window used for the estimation of the sample return covariance matrix based upon the current state of the environment. The resulting window size is then used to obtain the sample covariance matrix from which the weights of the GMV portfolio (as explained above) are analytically calculated and the investment is made accordingly. To further simplify the problem we make two additional assumptions. Specifically, the assumption of complete market liquidity which implies that each trade can be completed at the last observed price, and the assumption of zero trading costs.

III. LITERATURE REVIEW

Current research on the selection of covariance matrix estimation window size in the context of portfolio optimization is scarce, especially if adaptive approaches are considered. Bayraktar and Bilge [7] study optimal estimation widow sizes using classical Markowitz portfolio optimization, albeit in a non-adaptive fashion. Golosnoy [11] uses a time-varying window size for covariance matrix estimation in order to find the optimal trade-off between estimation error and bias. Optimal portfolio weights are monitored via the use of control charts that give an alarm when the control statistic leaves the acceptance area, indicating changes in the means of the optimal portfolio weights. The authors empirically show that time-varying window size strategies outperform the alternatives in most cases. Härdle et. al. [12] use a data-driven approach to adaptively vary the size of the window over which a local constant-parameter model is estimated for high-frequency financial variables. It is found that this adaptive approach yields significantly better forecasts than alternative approaches that use fixed-size estimation intervals. Finally, the authors conclude that adaptive approaches enable both better high-frequency forecasts and insights into local variations of the parameters.

IV. CONTRIBUTIONS

The main contributions of this paper include the following:

- We sucessfully use deep reinforcement learning for dynamic determination of the optimal size of the rolling window used for estimation of the sample return covariance matrix. To the best of the authors' knowledge, similar studies have not been previously done. The resulting agent achieves superior performance over various benchmark agents.
- We construct a novel state-space representation based on Frobenius norms of the differences between consecutive correlation matrix estimates in order to alleviate detection of correlation changes in the time series of financial returns. Although such approaches are commonly used for break-point detection in multivariate time series, this is, to the best of the authors' knowledge, the first attempt to combine such an approach with reinforcement learning to tackle the problem of rolling window length selection for portfolio optimization.

V. DATA AND MODEL DESIGN

A. Data

We consider a dataset comprised of daily closing stock prices from the New York Stock Exchange (NYSE), obtained via Yahoo! Finance. Firstly, we remove all stocks that include at least one missing value and randomly select a fixed subset of 20 stocks. The price data has a time span of nearly 29 years (from 1990-Jan-02 until 2018-Dec-04). The dataset is divided into three (training, validation and testing) sets with the approximate ratio of 0.60:0.20:0.20. After preprocessing, the data is fed into a neural network that acts as a policy network which should ideally output optimal actions for each set of inputs.

B. Reinforcement learning setting

We can represent the problem as a Markov Decision Process (MDP) where the agent (the investor) at each time step (a day) receives a trading signal (a function of current and past returns) from the environment (the market) and makes actions (the choice of the size of the rolling window used for estimation) based upon it. The environment then rewards the agent with a numerical reward (proportional to the negative variance of the realized portfolio returns). The sole goal of the agent is to find the optimal policy which maximizes the expected sum of discounted cumulative rewards (i.e. to minimize the variance of realized portfolio returns).

State space – In order to facilitate learning we first use raw returns to calculate sample correlation matrix estimates M_t with a fixed-size rolling window of 70 days for each day t from 1990-May-29 until 2018-Dec-04. We proceed by calculating the Frobenius norm F_t of the difference between matrices M_t and M_{t-1} for each day t from 1990-

May-30 until 2018-Dec-04. Finally, for each day t, we calculate the means of the sets $\{F^i\}_{i=0,\dots,29}$ where:

$$F_t^i = \{F_{t-21i}, F_{t-21i-1}, F_{t-21i-2}, \dots, F_{t-21i-20}\}$$

The obtained values $F^0, F^1, \dots, F^{29}$ are used to construct the state representation $S_t = \{F_t^0, F_t^1, \dots, F_t^{29}\}$ for each time t. Finally, we perform Z-score normalization on each of the state representation features. Intuitively, large entries in the state representation can be understood to imply correlation changes in the corresponding time periods which subsequently may lead to smaller optimal rolling window sizes.

Action space – Each action A corresponds to a certain choice of the size of the rolling window T used for estimation. To limit the number of available actions and hence enhance the training process, the rolling window size is given by $21(k+6)$ where $k \in \{0,\dots,24\}$. Therefore at any time t there are exactly 25 actions available to the agent, each one representing a rolling window of length $k+6$ in months (considering that the trading month consists of 21 days). Since t is in days, weights are rebalanced on a daily basis.

Rewards – All rewards R_i are set to 0 expect for the reward at the end which is proportional to the negative variance of the realized returns, thereby encouraging variance minimization. Therefore:

$$R_{T_e} = -\sum_{t=1}^{T_e} (r_t - \bar{r})^2$$

where T_e is the length of the episode, $\bar{r}$ the average portfolio return and r_t portfolio return at time t. Obviously, this is a case of sparse rewards. One should note that, since there is only one non-zero reward (precisely at the end of the episode), the discount rate is not relevant and can be set to 1.

Episodes – Each episode can be represented as a complete trajectory:

$$(S_0, A_0, R_0 = 0, \dots, S_{T_e}, A_{T_e}, R_{T_e} = -\sum_{t=1}^{T_e} (r_t - \bar{r})^2, S_{T_e+1})$$

Note that since we are backtesting (i.e. using historical data), transitions between states are deterministic and it generally holds that A_t does not influence S_{t+1}.

C. Policy (neural) network

The reinforcement learning agent is represented by a neural network. Two fully connected neural networks with different architectures are considered in model selection. Architecture A consists of only two hidden layers (with 512 and 256 neurons, respectively) while architecture B consists of five hidden layers (each one with 64 neurons). The ReLU function is used as an activation function in all hidden layers. The output layer in both architectures consists of a single neuron with a sigmoid activation function, and the final output is passed into to the appropriately scaled floor function. Finally, architecture A is chosen due to higher performance on the validation set. It should be noted that under this approach the neural network maps states directly into deterministic (optimal) actions instead of probability distributions over the action space. This is somewhat similar to the approach taken by Jiang and Liang [13] and has the advantage of eschewing the problem of high variance associated with more conventional approaches.

D. Optimization procedure

In order to train the weights of the neural network, a simple genetic algorithm is used. Genetic algorithms are a class of optimization algorithms suitable for „black-box" function optimization. Recently, they were shown to be competitive with gradient-based methods for deep reinforcement learning problems [14] [15]. Under such an approach, the function that maps the weights of the neural network to the expected reward (proportional to negative variance) is directly maximized. At the very start, a certain number of random agents are generated and evaluated. A number of top-performing agents are chosen as parents who yield offspring with the same weights up to a random perturbation. This process is then repeated for a large number of generations. To prevent overfitting, a variant of early stopping is employed. The exact hyperparameter values used for training are given in Table 1.

TABLE 1: GENETIC ALGORITHM - HYPERPARAMETER VALUES

Hyperparameter	Hyperparameter value
Number of agents per generation (population size)	50
Number of generations	300
Mutation rate	Decreases from 0.02 to 0.002
Mutation rate change frequency	once per 15 generations
Truncation size (number of top agents used as parents)	10

VI. RESULTS AND DISCUSSION

A. Simulated data

As a brief proof of concept, we train a neural network with a single hidden layer consisting of 10 neurons on a time series (length L=600) of returns generated by a five-dimensional multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}(t))$. Initially, $\boldsymbol{\Sigma}(t) = \boldsymbol{\Sigma}_1$. At the break point $t_1 = 100$ the covariance matrix abruptly changes to $\boldsymbol{\Sigma}_2$ ($\boldsymbol{\Sigma}_2 \neq \boldsymbol{\Sigma}_1$) and finally at $t_2 = 400$ it reverts back to $\boldsymbol{\Sigma}_1$. The mean $\boldsymbol{\mu}$ is constant throughout the episode. After training with multiple random seeds, the model is evaluated on a testing dataset consisting of a time series (again of length L=600) of returns generated by a different multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}'(t))$ with break points happening at $t_1' = 200$ and $t_2' = 500$. Figure 2 shows the agent's actions in the testing set. The agent seems to be able to react to both break points and adjust the estimation window size accordingly although with a significant lag ($\Delta t \approx 50$). Since the positions of break points were altered in the testing set, the results indicate that such an agent is, with the use of proper agent architecture and feature preprocessing, potentially capable of generalizing to unseen environments.

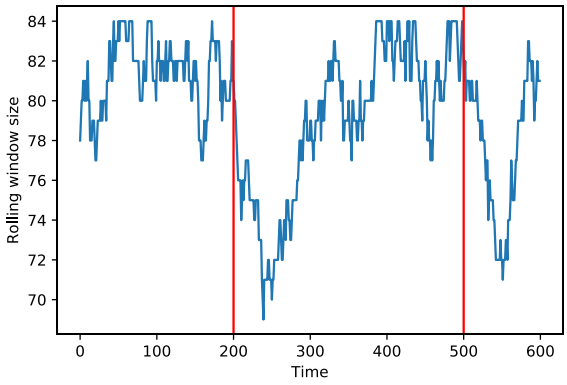

Figure 2: The agent's actions (choices of the rolling window sizes) in the testing set. Change points are denoted by vertical red lines.

B. *Real-world dataset*

After training, the model is evaluated on a testing dataset spanning the time period which lasts approximately 6 years. Figure 3 shows the agent's actions (chosen rolling window sizes) in the testing set as a function of time. The performance of the agent (with the ex-post annualized volatility used as the metric) is compared against traditional benchmarks (different choices of fixed sizes for rolling windows) and given in Table 2. Graphical comparison is made in Figure 4. The results clearly indicate that the agent successfully generalizes and manages to outperform the benchmarks, albeit quite modestly. Target downside deviations for the three portfolios are given in Table 3.

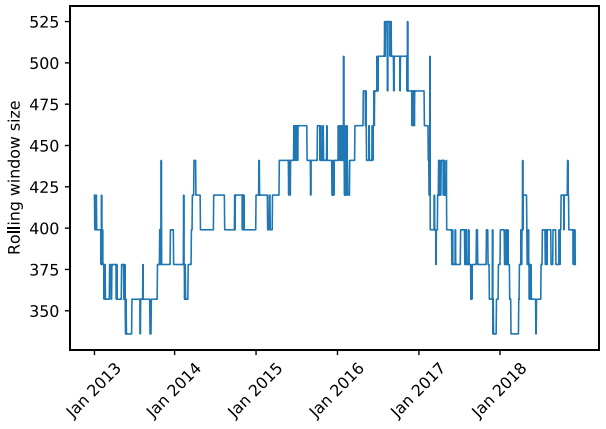

Figure 3: The agent's actions (choices of the rolling window sizes at each day t) in the testing set

TABLE 2: PORTFOLIO PERFORMANCE RESULTS MEASURED VIA ANNUALIZED EX-POST VOLATILITY

	Annualized ex-post volatility (assuming 252 trading days a year)
Our agent	10.91%
Best fixed window size (588, chosen *a posteriori*)	10.98%
Fixed window size (252, length of a trading year)	11.00%
Average random policy (100 iterations)	11.07%

TABLE 3: PORTFOLIO PERFORMANCE RESULTS MEASURED VIA TARGET DOWNSIDE DEVIATION

	Target Downside Deviation
Our agent	7.57%
Best fixed window size (588, chosen *a posteriori*)	7.59%
Fixed window size (252, length of a trading year)	7.63%
Average random policy (100 iterations)	7.65%

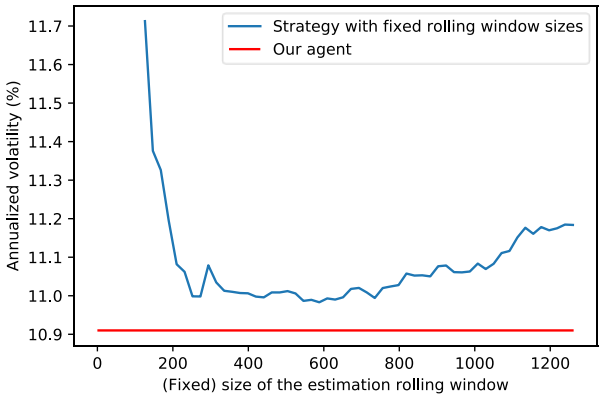

Figure 4: Comparison of annualized volatility achieved by our model versus annualized volatilities achieved by strategies using fixed rolling window sizes (in the testing set)

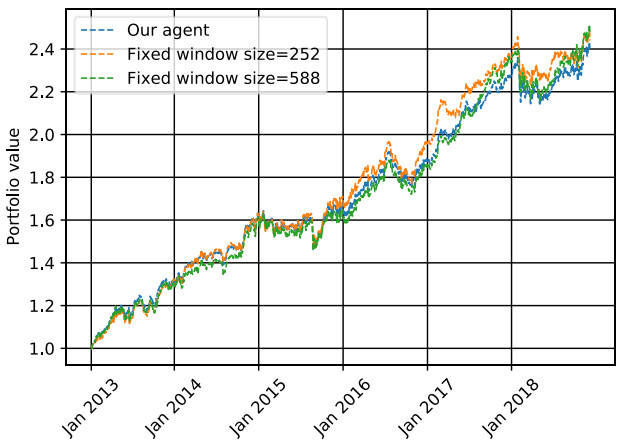

Figure 5: Portfolio value for both our agent and strategies using fixed rolling window sizes (in the testing set)

TABLE 4: PORTFOLIO PERFORMANCE RESULTS MEASURED VIA THE SHARPE RATIO

	Sharpe ratio (assuming 252 trading days a year)
Our agent	1.41
Best fixed window size (588, chosen *a posteriori*)	1.45
Fixed window size (252, length of a trading year)	1.43
Average random policy (100 iterations)	1.42

Additionally, Sharpe ratios are compared in order to determine if the obtained strategy is favorable when both risk and realized returns are accounted for. Note that the risk-free rate is set to 0 for the sake of simplicity. The performance results (shown in Table 4 and also in Figure 5) indicate that the resulting agent fails to attain a superior Sharpe ratio.

Although this is mostly unsurprising, considering our reward function is designed to only take variance into account, it is interesting to note that the results do not cast further doubt on the implications of the CAPM but are rather more aligned with them.

VII. LIMITATIONS AND FUTURE WORK

The main limitations include the assumption of no trading costs and the inclusion of only one type of state features. Furthermore, testing is performed on a period that is temporally distant from the training period, which implies that emphasis is given to long-term patterns between the state features and the reward, while shorter-term patterns are not captured by the model.

Among possible extensions of our work we mention enrichment of the state space with additional features (possibly macroeconomic indicators, interest rates data, yield curve data etc) and consideration of a wider spectrum of neural network architectures (especially convolutional neural networks). Additionally, instead of dynamically determining optimal discrete rolling window sizes, it would be interesting to perform a dynamic determination of optimal weighting schemes for covariance estimators (for instance, finding optimal smoothing factor values for exponential smoothing).

VIII. CONCLUSION

In this paper, we introduce an adaptive rolling window selection approach for minimum variance portfolio estimation and show that it manages to outperform the classical benchmarks (with fixed rolling window sizes) on a real-world dataset. The results suggest that using reinforcement learning based methods as an auxiliary tool in conjunction with more classical approaches, such as the analytic estimation of the GMV portfolio, may represent a feasible step toward better portfolio management strategies. It should also be noted that this paper is partly inspired by the line of research which includes alternative approaches to deep reinforcement learning based on genetic algorithms and evolutionary strategies which have been gaining traction lately due to their multiple favorable properties. Lastly, we emphasize that the outlined framework can be used with different policy architectures and is generally easily amenable to future enhancements.

IX. ACKNOWLEDGEMENTS

This work has been supported in part by Croatian Science Foundation under the project 5241 and in part by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

LITERATURE

[1] J. Philippe, “Bayesian and CAPM estimators of the means: implications for portfolio selection,” Journal of Banking and Finance 15, no. 3, pp. 717–727, June 1991.

[2] R. Clarke, H. De Silva, and S. Thorley, "Minimum-variance portfolio composition," The Journal of Portfolio Management 37, no. 2, pp. 31-45, January 2011.

[3] R. Clarke, H. De Silva, and S. Thorley, "Minimum-variance portfolios in the US equity market," The Journal of Portfolio Management 33, no. 1, pp. 10-24., October 2006.

[4] M. Baker, B. Bradley and J. Wurgler, „Benchmarks as limits to arbitrage: Understanding the low-volatility anomaly,“ Financial Analysts Journal 67, no. 1, pp. 40-54., January 2011.

[5] R. C. Merton, „On estimating the expected return on the market: an exploratory investigation,“ Journal of Financial Economics 8, no. 4, pp. 323-361., December 1980.

[6] B. Scherer, “A new look at minimum variance investing,” Journal of Empirical Finance 18, no. 4, pp. 652-660., July 2010.

[7] E. Bayraktar and A. H. Bilge, “Determination the parameters of Markowitz portfolio optimization model,” International Conference on Mathematical Finance and Economics (ICMFE) 2011.

[8] O. Lediot and M. Wolf, „Improved estimation of the covariance matrix of stock returns with an application to portfolio selection,“ Journal of Empirical Finance 10, no. 5, pp. 603-621, December 2003.

[9] E. Pantaleo, M. Tumminello, F. Lillo and R. N. Mantegna, „When do improved covariance matrix estimators enhance portfolio optimization? An empirical comparative study of nine estimators,“ Quantitative Finance 11, no. 7, pp. 1067-1080., July 2011.

[10] S. Begušić and Z. Kostanjčar, „Cluster-based shrinkage of correlation matrices for portfolio optimization“, 11th International Symposium on Image and Signal Processing and Analysis (ISPA), pp. 301-305., IEEE, September 2019.

[11] V. Golosnoy, „Sequential monitoring of minimum variance portfolio,“, AStA Advances in Statistical Analysis 91, no. 1, pp. 39-55., March 2007.

[12] W. K. Härdle, N. Hautsch, and A. Mihoci, "Local adaptive multiplicative error models for high‐frequency forecasts," Journal of Applied Econometrics 30, no. 4, pp. 529-550., June 2015.

[13] Z. Jiang and J. Liang, „Cryptocurrency portfolio management with deep reinforcement learning,“ 2017 Intelligent Systems Conference (IntelliSys), pp. 905-913.. IEEE, September 2017.

[14] F. Such, et al., „Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning,“, NeurIPS Deep Reinforcement Learning Workshop, 2018.

[15] A. Sehgal, H. La, S. Louis and H. Nguyen, „Deep reinforcement learning using genetic algorithm for parameter optimization“, 2019 Third IEEE International Conference on Robotic Computing (IRC), pp. 596-601., IEEE, 2019