CS230

Time is of the essence: Portfolio multi-scale scheduling via Reinforcement Learning

Emile Clastres
Stanford University
clastres@stanford.edu

Abstract

Portfolio optimization papers overwhelmingly overlook scheduling when designing strategies, and choose to adopt a calendar rebalancing schedule, recomputing portfolio weights at fixed time intervals using a fixed-size time window for weight computation. But this simple approach lacks adaptability when the rate at which the market evolves is not constant over time. In previous work, we have shown that the choice of a rebalancing schedule can have a dramatic impact on performance. We have proposed the mu-sigma schedule, a rule-based dynamic alternative to calendar rebalancing that builds on recent findings in Random Matrix Theory to replace time with covariance distance as the fundamental scheduling metric, and improves the Sharpe Ratio of strategies significantly. In this work, we learn data-driven schedules based on Reinforcement Learning and Distance Scheduling, that should maximize the performance of portfolios while bridging the gap between intra-day and low-frequency portfolio strategies. Our learned schedules bring up to 50% improvement in Sharpe Ratio, and are the first financial machine learning models to be able to generalize across markets and sampling frequency.

1 Introduction

Portfolio allocation is often considered to have been fathered by Markowitz's mean-variance framework in the 50's (1). Markowitz's methods and its offspring aim at maximizing expected returns for a given level of risk. In other words, mean-variance portfolio allocation methods all try to tackle the mean-variance trade-off in an optimal way. In practice, future returns are unknown to the investor while risk is subject to non-stationarity and a high estimation error, leading to under-performing portfolios out of sample (2).

This issue led to risk-based strategies, focused entirely on minimizing portfolio risk, such as the Global Minimum Variance Portfolio (3). Other authors have focused on maximizing diversification criteria other than risk (4; 5).

Once a portfolio allocation method is selected, one must choose when to rebalance portfolio weights. However, market fees associated with these transactions lead to a trade off between responsiveness and transaction costs – called the information/fee trade-off hereafter. For this matter, the most widely adopted schedule is calendar rebalancing, which defines a fixed time period T for rebalancing. This method is notoriously weak due to its lack of adaptability. Indeed, as T is independent of the rate at which the market evolves it will become relatively too large during high volatility periods and too small otherwise. Yet, it is widely used in the literature to benchmark strategies and it constitutes the foundation of more sophisticated schedules.

CS230: Deep Learning, Fall , Stanford University, CA.

As all previously mentioned allocation methods are functions of an estimate of the covariance matrix, our previous work (6) proposed to use a measure of distance in the space of positive semi-definite matrices to choose when to rebalance the portfolio. Put differently, instead of waiting for T days before rebalancing the portfolio, we wait for the current covariance matrix to exceed a certain distance d relative to the covariance matrix at the time of the last rebalancing operation. Such a distance estimation would have been too unstable and erroneous in the usual setting, but recent advances in Random Matrix Theory (7) have made covariance matrix distance estimation more reliable by orders of magnitude in the typically commensurably large m, N (number of assets, number of samples) regime of financial time series. We introduced a dynamic distance-based rebalancing scheme based on these considerations which was shown to consistently reduce turnover and increase the Sharpe Ratio of allocation methods by a significant margin independently of the allocation method selected.

In this work, we propose to directly learn optimal schedules via Reinforcement Learning. The use of RL in portfolio allocation has been focused on the determination of optimal portfolio weights (8; 9). This task has proved very challenging, and most current attempts don't scale to large portfolios because of dimensionality explosion in both the observation and action spaces. On the other hand, our proposed task of learning schedules is an easier – yet never attempted – task. The agent has to learn when it is worth it to pay transaction fees and update its allocation with the latest market information, which is an action space independent of the portfolio size. Another advantage of such an approach is that it allows the investor to improve all its allocation methods, and thus benefits from all his previous work. Furthermore, there is a lesser risk of unanticipated black-box failure since the weights are still computed using the allocation method's rationale, and the agent only learns the timing of updates – a behavior easily monitored and constrained if need be.

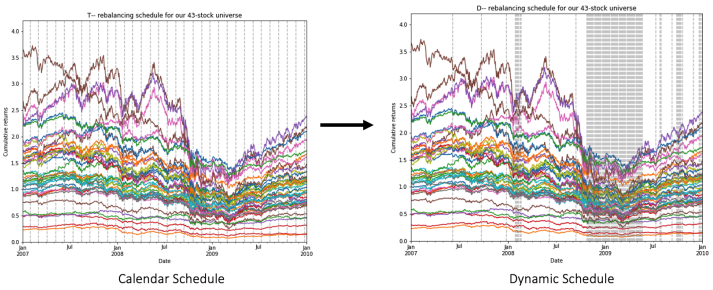

Figure 1: Schedule examples, vertical dotted lines represent reallocation dates.

2 Related work

Reinforcement Learning has been considered for trading for a long time, due to the inherently multi-stage decision process of a trading strategy. In 1998 already, Moody (8) proposed a Reinfircement Learning framework for trading, and proposed an adequate reward function, the differential sharpe ratio, which allows for the on-line maximimization of the sharpe ratio. Yet, these approaches suffered from the dimensionality of the observation space, linear in the size of he investment universe, as well as the action space.

Advances in Deep Reinforcement Learning gave rise to two different approaches to the problem. The first approach is to learn single-asset strategies directly via model-free Deep Reinforcement Learning (9). This method has shown to be promising, however it relies on the possibility to predict future stock returns, which is considered hardly possible. The second approach is to construct portfolios directly (9; 10; 11). These approaches feed market data to the agent, which has to output a vector of allocation weights w for every time step. This problem is paradoxically easier conceptually, because the agent can get performance from the combination of stocks. Fundamentally, there are simple combinations of stocks that allow to hedge risk and get good performance, and the agent can thus obtain performance without relying on the accurate prediction of each individual stock price. Yet, when a reasonable number of stocks (say, 50) is provided, the observation and action spaces become too large for learning to take place with the relatively meagre data available. It is all the truer since these systems have to be trained on recent data to be relevant, which restricts the amount available. To tackle this issue, (12) proposed the use of Ensemble of Identical and Independent Estimators, which

relies on weight sharing to output an individual score for each stock, then the scores are combined in a shallow fashion. Reccurent Topologies such as LSTM were also considered and proved more powerful, yet more difficult to train. With the exception of (12), all these frameworks couldn't scale to more than a dozen assets.

The Scheduling problem proposed here doesn't suffer from the same difficulties, because it is independent on the size of the market in both observation and action. It was first introduced in our previous work (in progress) (6). In this work, it was shown that a dynamic schedule based on distance instead of time could outperform the calendar schedule by up to 20% in Sharpe Ratio, without being conceptually more complex.

3 Methods

3.1 Preliminaries: Scheduling Framework

Consider the returns matrix of m assets over N timestamps ($\mathbf{X} \in \mathcal{M}_{m,N}$). We need to choose the dates $(t_n)_{n\in\{1...p\}}$ at which we rebalance our portfolio weights $(\mathbf{w}_n)_{n\in\{1...p\}}$. We call *schedule* such an increasing sequence $(t_n)_{n\in\{1...p\}}$. In essence, what we need is a scheduling function f such that[1]:

$$f(t_n, \mathbf{w}_n, \mathbf{X}_{[0:t]}, t) = \begin{cases} 0 \text{ if } t < t_{n+1} \\ 1 \text{ if } t \geq t_{n+1} \end{cases}$$

That way, we can build schedules by setting $t_0 = N$ and $\forall n \geq 1, t_{n+1} = \min\{t \geq t_n | f(t_n, \mathbf{w}_n, \mathbf{X}_{[0:t]}, t) = 1\}$.

The calendar schedule of period T can be easily expressed in this setting with $f(t_n, \mathbf{w}_n, \mathbf{X}_{[0:t]}, t) = \textit{sign}(t - t_n - T)$, with convention $sign(0) = 1$. Most investors schedules incorporate conditions such as concentration limits (e.g. CPPI). All these rules can be expressed as additional conditions on schedules with *or*, *and* operators. In what follows, we attempt to learn a scheduling function. This function will make use of Covariance distance, which is the fisher distance between the sample covariance matrix at time t and the covariance matrix at time t_n. There are important considerations when computing distances in the regime of financial time series, which are discussed in the appendix.

3.2 Environment

Our problem revolves around the concept of ideal portfolio. Given an allocation method, the ideal portfolio is the portfolio that maximizes the Information / Friction tradeoff. In other words, it is the portfolio obtained by rebalancing as frequently as possible, while paying no fees. Our agent must learn to best track the ideal portfolio while not being exempt from taxes.

As a result, its reward function is the difference between the differential sharpe ratio of the traded portfolio and the differential sharpe ratio of the ideal portfolio. In practice, it is possible to outperform the ideal portfolio due to the imperfect nature of the allocation method. However, it is not desirable for our agent to pick up these anomalies, and therefore we cap its rewards at 0, meaning that it will not be rewarded for the faiure of the allocatio method. Coversely, this difference makes it so that the rewards of the agent are independent on the underlying market. The agent is rewarded only for not underperforming the ideal portfolio. This removes some stochasticity to the environment and makes generalization possible.

The features are four-fold.

- Path dependant Distance-based features : These are the distance between the current oberved sample covariance matrix and the covariance matrix used for the last update. These features are normalize through a rolling exponential moving z-score, using pools of observation fro shifted covariance distances of different scales.
- Time-based features : these features are binary numbers representing wether or not more than T steps have elapsed since the last update.

[1] Schedule functions will often be defined without respecting the monotonicity constraint. In practice, any boolean function can be made non-decreasing by using cumulative **or** statements.

- Tracking features: These track the performance relative to the ideal portfolio. We use the z-score of the variance of the difference of the returns between the actual portfolio and the ideal portfolio.
- Virtual turnover: This is the turnover the agent would pay at step t for rebalancing. It allows the agent to avoid updating when the update contains a high noise.

These features are accessed with lookbacks to allow temporal decisions dependent on previous states.

The actions are binary rebalancing decisions : 1 for rebalancing, 0 for keeping current weights.

The model used is DQN, which is particularly sample efficient. This is an important property for compute budget reasons. Moreoever, its deterministic nature allows to evaluate performance on single runs.

4 Experiments/Results/Discussion

This work aims to be as general as possible. Therefore, the only data used will be stock returns data. We use a dataset comprising 43 daily prices European Indexes from 2000 to 2019. The results are sensitive to the choice of the discount factor. If the discount factor is too high, then the agent doens't learn that he is paying transaction fees. On the other hand, a too low discount will make the agent learn to never rebalance because he will be focused on the immediate cost of trading. A good compromise is $\gamma = 0.3$, used hereafter. We train the agent using three different timescales for distance estimation, time thresholds of 5, 20 and 80 steps, and lookbacks of 1,2,3,4,5 and 40 steps. Multiple experiments have shown that the choice of these parameters only have minor impact on the ability for the agent to learn schedules. The agent is trained for 50,000 steps, which corresponds to roughly 10 runs of the whole dataset. The allocation method used by the agent is the Global Minimum Variance Portfolio, chosen for its simplicity and because it is computationally inexpensive.

The agent is trained using the 3000 first timesteps of the dataset, and evaluated out of sample on the last 1700. Figure 2 represents the results for the agent out-of-sample. It obtains a Sharpe Ratio 15% better than the calendar schedule (parametrized to rebalance every 4 weeks, a standard for this dataset). Moreover, it outperforms the distance-based mu-sigma schedule introduced in (6) by 4% with a similar number of rebalancing operations.

In-sample (Figure 3), the overperformance in Sharpe Ratio over the Calendar Schedule is as high as 60%, with a very reasonable schedule which rebalances frequently during market shifts and sparsely otherwise. This extreme overperformance doesn't look to be the result from overfitting but rather because of the presence of the 2008 financial crisis in the In-Sample period, which constitutes great potential for dynamic schedules to outperform calendar schedules.

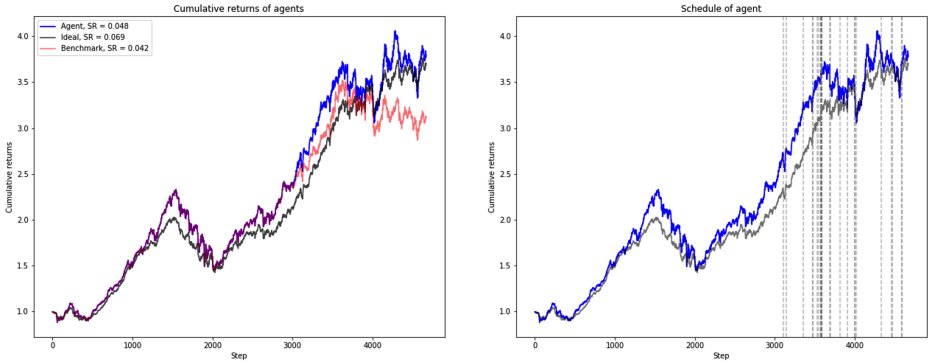

Figure 2: Scheduling Performance Out-of-Sample

An interesting aspect of this approach is that learning can take place across different markets, and different time periods. One can use and apply a model to different markets, thus multiplying the amount of data it has seen. This is of special interest since scheduling becomes especially important during exceptional periods like financial crises. Therefore, it would benefit a lot from having been trained on different markets with different crises.

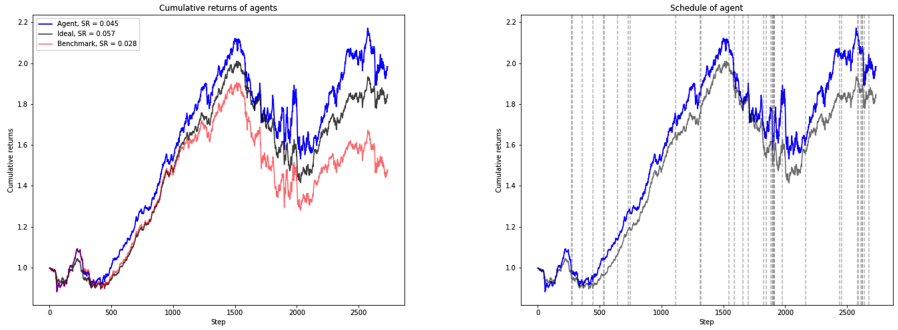

Figure 3: Scheduling Performance In-Sample

Yet, learning is actually very robust by design. Our simple DQN agent was tested on a absolutely different market made of 12 cryptocurrencies (instead of 43 indexes), on a different time period (2020), with even a different frequency (hourly returns instead of daily). In these extreme conditions, it is still able to outperform the benchmark by 40% which is the calendar schedule with daily operations (which happens to be the frequency used by the agent). To the best of our knowledge, this is the first time an agent presents such generalization in finance. The produces schedule is also reasonable, with high frequency rebalancing when the market undergoes rapid transformations, and more stable operations otherwise. Results are visible in Figure **??** showcases the results on the extreme generalization test.

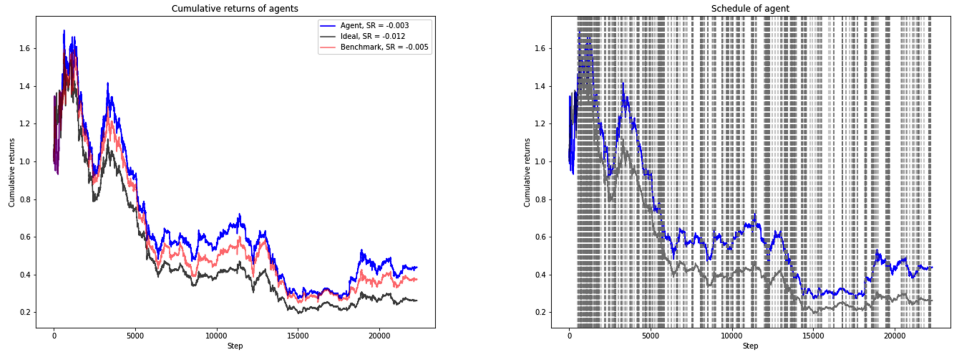

Figure 4: Scheduling Performance on Cryptocurrency universe

5 Conclusion/Future work

Scheduling is fertile grounds for portfolio improvement. Not only is it applicable to any allocation method including legacy methods, but it constitutes a tractable problem fundamentally less sensitive to the underlying specifics of the market. It is scalable, robust and general and it allows improvement in Sharpe Ratio that are akin to actual changes of methods. It is all the more important since in previous investigations, we have found that the choice of a schedule changes the ranking of different allocation strategies, meaning that benchmarking allocations on a common schedule is unfair and unscientific. Rather, portfolio optimization researchers should evaluate each method with an adapted schedule, possibly learned.

In this work, simple Deep Reinforcement learning agents were able to successfully learn schedules that outperform out-of-sample, even in extreme generalization test that were never attempted for Machine Learning models in finance. Future work will involve the introduction of appropriate Recurrent agents that will further reduce the observation space. Moreover, one could consider continuous actions, which would rebalance only a fraction of the portfolio. These fractional updates would allow more flexibility especially in hourly or minute frequency markets, where updates are too

rare (1 step out of 10000) to be learned effectively. This will require new work in feature design, since a fractional control agent can not use metrics relative to the "last operation" since it is undefined.

Lastly, the choice of a time window to estimate the covariance matrix is also a degree of freedom that can be explored. Undoubtedly, the joint choice of timing and temporality can bring even more value to allocation methods, which would effectively be usable on a true multi-scale fashion. Minute-frequency data could be used to produce portfolios able to adapt to different temporalities without being restricted by design.

A Random Matrix Theory Distance Estimation

Let $\mathbf{X}_1, \mathbf{X}_2 \in \mathcal{M}_{m,N}$ be two matrices of m asset returns over N timestamps. These matrices are supposed to be jointly i.i.d. realizations of random processes with covariance matrices $\mathbf{C}_1$ and $\mathbf{C}_2$. We further let $\hat{\mathbf{C}}_1 = \frac{1}{N}\mathbf{X}_1^T\mathbf{X}_1$ and $\hat{\mathbf{C}}_2 = \frac{1}{N}\mathbf{X}_2^T\mathbf{X}_2$ be their respective in-sample covariance matrices (SCM). In order to determine whether or not rebalancing should take place at a given time instant, distance-based rebalancing strategies (introduced in this paper) require to evaluate some distance between the covariance matrices $\mathbf{C}_1$ of the returns $\mathbf{X}_1$ in the last allocation and $\mathbf{C}_2$ of the current returns $\mathbf{X}_2$.

Covariance matrices being positive definite, the Fisher distance $D_F(\mathbf{C}_1, \mathbf{C}_2)$ between $\mathbf{C}_1$ and $\mathbf{C}_2$ is particularly adapted: it corresponds to the length of the geodesic joining $\mathbf{C}_1$ and $\mathbf{C}_2$ in the space $\mathcal{S}_m$ of positive semi-definite matrices of size m and reads

$$D_F(\mathbf{C}_1, \mathbf{C}_2) = \frac{1}{m}\| \log(\mathbf{C}_1^{-\frac{1}{2}}\mathbf{C}_2\mathbf{C}_1^{-\frac{1}{2}})\|_F^2 = \frac{1}{m}\sum_{i=1}^{m} \log^2\left(\lambda_i(\mathbf{C}_1^{-1}\mathbf{C}_2)\right)$$

with $\|\cdot\|_F$ the Frobenius norm, $\mathbf{C}^{\frac{1}{2}}$ the nonnegative definite square root of $\mathbf{C} \in \mathcal{S}_m$ and $\log(\mathbf{C}) = \mathbf{U}\log(\mathbf{\Lambda})\mathbf{U}^T$ where $U\Lambda U^T$ is the spectral decomposition of $\mathbf{C} \in \mathcal{S}_m$.

In practice, $\mathbf{C}_1$ nor $\mathbf{C}_2$ being unknown, one needs to produce an estimate of $\hat{D_F}(\mathbf{C}_1, \mathbf{C}_2)$ based on the samples $\mathbf{X}_1$ and $\mathbf{X}_2$ which, in the present setting may be such that the number of available samples N is not necessarily much larger than the data dimension m.

The simplest, and most conventionally used, estimator is the plug-in estimator, which consists in replacing (plugging) the unknown covariance matrices by its in-sample estimate (SCM) in the formula for the Fisher distance, i.e.,

$$\hat{D}_F^{\mathrm{SCM}}(\mathbf{X}_1, \mathbf{X}_2) \equiv D_F(\hat{\mathbf{C}}_1, \hat{\mathbf{C}}_2).$$

It is in particular not difficult to ensure that $\hat{D}_F^{\mathrm{SCM}}(\mathbf{X}_1, \mathbf{X}_2) \to D_F(\mathbf{C}_1, \mathbf{C}_2)$, almost surely, as $N \to \infty$, while m remains fixed. However, under the setting where $m, N \to \infty$ such that $c \equiv \frac{m}{N}$ remains in $(0, 1)$ (our present setting), the operator norm $\|\hat{\mathbf{C}} - \mathbf{C}\|$ no longer vanishes and, as a result, the plug-in estimator becomes inconsistent (7).

As a counter-measure, (7, Corollary 4) provides an "m, N"-consistent estimator of distance for D_F, as follows.

Proposition 1 (m, N-consistent estimator of D_F)**.** *Under the present setting and with $\|\mathbf{C}_1\|$, $\|\mathbf{C}_2\|$ bounded with m, as $m, N \to \infty$ with $c \equiv m/n \in (0, 1)$ remaining away from 0 and 1,*

$$D_F^{\mathrm{RMT}}(\mathbf{X}_1, \mathbf{X}_2) - D_F(\mathbf{C}_1, \mathbf{C}_2) \to 0$$

almost surely, where

$$D_F^{\mathrm{RMT}}(\hat{\mathbf{C}}_1, \hat{\mathbf{C}}_2) \equiv \left[\frac{1}{m}\sum_{i=1}^{m} \log^2\left((1-c)\,\lambda_i\right) + 2\frac{2c - c^2}{c^2}\left\{\left(\Delta_\zeta^\eta\right)^\top \mathbf{M}\left(\Delta_\lambda^\eta\right) + \left(\Delta_\lambda^\eta\right)^\top r\right\}\right.$$
$$\left. - \frac{2}{m}\left(\Delta_\zeta^\eta\right)^\top \mathbf{N}1_m - 2\frac{1-c}{c}\left\{\frac{1}{2}\log^2\left((1-c)^2\right) + \left(\Delta_\zeta^\eta\right)^\top r\right\}\right]$$

where $\Lambda = \mathrm{diag}(\lambda)$ *for* $\lambda = (\lambda_1, ..., \lambda_m)^T$ *and* $0 < \lambda_1 < \ldots < \lambda_m$ *the eigenvalues of* $\hat{\mathbf{C}}_1^{-1}\hat{\mathbf{C}}_2$ *sorted in ascending order;* $\sqrt{\lambda} = (\sqrt{\lambda_1}, ..., \sqrt{\lambda_m})^T$. *Similarly,* $0 < \eta_1 < \ldots < \eta_m$ *are the eigenvalues of* $\Lambda - \frac{\sqrt{\lambda}\sqrt{\lambda}}{m-N}$ *and* $0 < \zeta_1 < \ldots < \zeta_m$ *the eigenvalues of* $\Lambda - \frac{\sqrt{\lambda}\sqrt{\lambda}}{N}$*;* Δ_a^b *is the vector with* $\left(\Delta_a^b\right)_i = b_i - a_i$ *and, for* $i, j \in \{1, \ldots, m\}$, $r_i = \frac{\log((1-c)\lambda_i)}{\lambda_i}$ *and*

$$\mathbf{M}_{ij} = \begin{cases} \frac{\frac{\lambda_i}{\lambda_j} - 1 - \log\left(\frac{\lambda_i}{\lambda_j}\right)}{(\lambda_i - \lambda_j)^2} & , \, i \neq j \\ \frac{1}{2\lambda_i^2} & , \, i = j \end{cases}, \quad \mathbf{N}_{ij} = \begin{cases} \frac{\log\left(\frac{\lambda_i}{\lambda_j}\right)}{\lambda_i - \lambda_j} & , \, i \neq j \\ \frac{1}{\lambda_i} & , \, i = j. \end{cases}$$

With these elements in place, we are in position to introduce our considered setting and our proposed features for portfolio scheduling.

B Modeling and practical distance estimation

Let us consider a typical portfolio allocation setting. We wish to manage a portfolio of m indices (m is, say, typically of order 50), using a sliding covariance matrix estimation window of size N (say, of order 260 days amounting to a full year of daily returns). For notational convenience, $\hat{\mathbf{C}}_{[t:t+N]}$ stands for the SCM computed with returns $\mathbf{X}_{t:t+N}$ from date t to date $t + N - 1$. In order to choose when to rebalance the portfolio, our proposed strategy is to evaluate the covariance matrix distance $D_F(\hat{\mathbf{C}}_{[t:t+N]}, \hat{\mathbf{C}}_{[t+k:t+N+k]})$ for all $k \geq 1$ until a "loss of stationarity" threshold is exceeded. In practice, if t denotes the date of the last rebalancing operation, this corresponds to computing every day the distance between the current observed covariance matrix and the covariance matrix used at time t.

Two practical considerations need be addressed in order to estimate $D_F(\hat{\mathbf{C}}_{[t:t+N]}, \hat{\mathbf{C}}_{[t+k:t+N+k]})$,

1. in order to apply Proposition 1, one demands that $\mathbf{X}_{t:t+N}$ and $\mathbf{X}_{t+k:t+k+N}$ be independent and thus non-overlapping. But, for $k < N$, both datasets share a proportion $\frac{N-k}{N}$ of their samples. This is all the more critical that the window used for covariance estimation is typically one order of magnitude larger than the time between two rebalancing operations (the general practice is that rebalancing takes place every few weeks and covariance estimations use a year of daily returns, as in (13; 3; 4; 14) etc.). To address this issue, we proceed by using instead $\hat{\mathbf{C}}_{[t:t+N]:\mathbf{2}}$ and $\hat{\mathbf{C}}_{[t+k:t+N+k]:\mathbf{2}}$, computed using only every other sample (at least in the overlapping time period), with the requirement that they share none of them. As a consequence, independence is granted at the cost of dropping up to one half of the samples, and instead of exploiting a dimension-to-sample ratio $c = \frac{m}{N}$, the latter is reduced to $c \approx \frac{m}{2N}$, further increasing the need for a RMT-based consistent distance estimator.
2. there may be situations in which investors want to manage a number of assets m greater than the estimation window N they use. In this case, there is a strong violation of the $c < 1$ assumptions. As a consequence of Item 1, a representative collection of asset returns smaller than $\frac{N}{2}$ should thus be used for distance estimation.

References

[1] H. Markowitz, "Portfolio selection*," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[2] R. O. Michaud, "The markowitz optimization enigma: is 'optimized' optimal?," *SSRN Electronic Journal*, 1989.

[3] R. Clarke, H. de Silva, and S. Thorley, "Risk parity, maximum diversification, and minimum variance: An analytic perspective," *SSRN Electronic Journal*, 2012.

[4] Y. Choueifaty and Y. Coignard, "Toward maximum diversification," *The Journal of Portfolio Management*, vol. 35, pp. 40–51, Oct. 2008.

[5] Y. Choueifaty, T. Froidure, and J. Reynier, "Properties of the most diversified portfolio," *SSRN Electronic Journal*, 2011.

[6] E. Clastres, "Improving portfolio rebalancing schedules with random matrix theory," 2020.

[7] R. Couillet, M. Tiomoko, S. Zozor, and E. Moisan, "Random matrix-improved estimation of covariance matrix distances," 2018.

[8] J. Moody and M. Saffell, "Reinforcement learning for trading," *Advances in Neural Information Processing Systems*, vol. 11, pp. 917–923, 1998.

[9] A. Filos, "Reinforcement learning for portfolio management," *arXiv preprint arXiv:1909.09571*, 2019.

[10] Z. Jiang and J. Liang, "Cryptocurrency portfolio management with deep reinforcement learning," in *2017 Intelligent Systems Conference (IntelliSys)*, pp. 905–913, 2017.

[11] Z. Zhang, S. Zohren, and S. Roberts, "Deep reinforcement learning for trading," *The Journal of Financial Data Science*, vol. 2, no. 2, pp. 25–40, 2020.

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

[13] M. L. de Prado, "Building diversified portfolios that outperform out of sample," *The Journal of Portfolio Management*, vol. 42, pp. 59–69, May 2016.

[14] L. Yang, R. Couillet, and M. R. McKay, "A robust statistics approach to minimum variance portfolio optimization," *IEEE Transactions on Signal Processing*, vol. 63, pp. 6684–6697, Dec. 2015.