Dynamic Portfolio Optimization Using Evolution Strategy

Kexin Yu
Stanford University
kexinyu@stanford.edu

Charles Xu
Google, Inc.
chx@google.com

***Abstract*— We devise an Evolution Strategy (ES) for the multi-asset multi-period portfolio optimization under both transaction costs and non-linear, non-convex risk constraints such as Value-at-risk and Expected Shortfall. Unlike the traditional mean-variance approach that makes decisions only for a single upcoming period, we aim at a more dynamic strategy in which the portfolio is rebalanced at different points of the holding period. To accomplish this, we generate a scenario tree which models the randomness of the time paths of asset prices based on historical data. Optimized with ES, every single path in the tree suggests a unique sequence of portfolios that meets customized financial goals. In addition, our model accommodates buy/sell transaction costs and ensures the admissibility of proposed strategies.**

I. INTRODUCTION

Portfolio management is the problem that given an investment time horizon and a list of assets with their historical prices, allocate for each asset a percentage weight in the portfolio such that the total risk-adjusted return is optimized. For decades, one classic approach to portfolio optimization is to maximize the Sharpe ratio of the portfolio, which is the expected return over its variance [1, 2]. Such mean-variance method has proven suboptimal since summary of history is used directly as prediction and asset weights remain constant for an extended period. Moreover, variance is not an ideal risk measure since it penalizes not only negative but also positive shocks and shows little about the likelihood and magnitude of tail risks [3]. Other risk measures, such as Value-at-risk and Expected Shortfall, have been proposed but introduced non-linear, non-convex risk constraints to the portfolio optimization problem [4, 5], which renders the mean-variance approach less applicable.

We see room for improvement by using Evolution Strategy (ES). Time series such as asset prices often violate the assumption of independent sampling and homoscedasticity [6]. Unlike most models introduced in class, ES makes no such assumption. Moreover, ES overcomes many inconveniences of other popular techniques such as neural network and reinforcement learning. The algorithm explores the parameter space by repeatedly injecting Gaussian noise in the current guess, generating a new population of candidates, and moving it towards dominant traits and strategies, which requires no backpropagation. Since each candidate can be evaluated independently, ES is also highly parallelizable [7].

Leveraging this technique, we adjust the asset weights as we go (hence dynamic optimization) in the hope for significantly higher returns under complex constraints in risks, positive weights, and disproportional transaction costs.

II. BACKGROUND

A. The Dynamic Portfolio Optimization Problem

Suppose there are M assets in the portfolio. Let $x = (x_1, ..., x_M)^T$ denote the M-dimensional weight vector where x_i is the weight allocated to asset i. Let $\xi = (\xi_1, ..., \xi_M)^T$ denote the prices vector where ξ_i is the price of asset i. Let $x^{(t)}$ and $\xi^{(t)}$ be observations at time t. The portfolio value at time t is therefore $W^{(t)} = {x^{(t)}}^T \xi^{(t)}$. The objective is to maximize $W^{(T)}$ by adjusting $\{x^{(t)} : T_0 \leq t < T\}$ given only $\{\xi^{(t)} : t \leq T_0\}$, subject to the following constraints:

1) Equal Wealth across Re-balancing:

$$ {x^{(t-1)}}^T \xi^{(t)} - C(x^{(t-1)}, x^{(t)}, \xi) = {x^{(t)}}^T \xi^{(t)} \ \forall t \quad (1) $$

where C denotes the transaction cost of entering or unwinding a position.

2) No Short Position:

$$ x_i^{(t)} \geq 0 \ \forall i, t \quad (2) $$

3) Value at Risk:

$$ VaR_\alpha \geq K \quad (3) $$

where VaR_α is defined as

$$ \Pr(W^{(T)} < VaR_\alpha) = \alpha \quad (4) $$

and K and α are enforced by regulators or portfolio managers. It is a risk constraint that says there is at most an α chance that the portfolio value at time T will end up less than K. Unlike variance as a risk measure, value-at-risk concerns only about the downside and does not penalize positive shocks.

4) Expected Shortfall:

$$ ES_\alpha \geq S \quad (5) $$

where ES_α is defined as

$$ ES_\alpha = \mathbb{E}[W^{(T)} | W^{(T)} < VaR_\alpha] \quad (6) $$

and S and α are enforced by regulators or portfolio managers. Expected shortfall shows the magnitude of tail risk. It is a risk constraint that says given VaR_α has been violated, the expected value of portfolio at time T should be at least S.

B. Assumptions

We assume there are 252 trading days annually and 21 trading days monthly.

III. DATA AND FEATURE

We have built and open-sourced[1] a tool that sweeps across Nasdaq and NYSE to fetch for each ticker its entire historical daily prices, record to durable storage and use as features. It coordinates a pool of workers with remote procedure calls (RPCs) using gRPC framework and protocol buffer as IDL (interface description language) and serialization protocol.

IV. METHODS

A. Tree Structure Interpretation

In our model, we construct an ordered, directed scenario tree to approximate the evolution of a portfolio over multiple time periods. [8] Each level of the tree is associated with asset prices at a different point of the holding period. Each node, with its portfolio weight vector, corresponds to a specific strategy at that given time. The root node starts with an initial capital and the goal is to maximize the final wealth realized by the leaf nodes. The binary structure of the tree makes it easy to backtrack the unique path leading to each leaf.

B. Design of Evolution Strategy

The algorithm can be summarized as follows:

1) Initialization: Suppose there are N nodes in the tree. The parent node k of a node n is $k = \pi(n)$. Random initialization is performed stepwise starting at the root node with an initial capital W_0 and must fulfill the *equal wealth* constraint in (1), in particular,

$$W_0 = x_0^T \xi_0 \tag{7}$$

$$x_k^T \xi_k + C(x_{\pi(k)}, x_k, \xi_k) = x_{\pi(k)}^T \xi_k \quad \forall k = 1, ..., N-1 \tag{8}$$

and the *no short position* constraint in (2) as well.

2) Mutation: Similarly, mutation is done by traversing the tree. In each generation, we perturb the weights by adding random noise drawn from the Normal distribution. To fulfill the *equal wealth* constraint, we fix $(M-1)$ components in the weight vector and derive the remaining one from (1). However, the resulting weight vector may violate the *no short position* constraint. To resolve this, we correct the negative component by projecting onto the positive orthant which minimally changes the original vector.

3) Selection: We choose the (μ, λ)-selection scheme. In each generation, μ parents breed λ offspring individuals, out of which the fittest μ are used as parents for the next generation. (Here, each offspring individual represents a new possible tree pattern with Gaussian noise.) This scheme is preferred over the $(\mu + \lambda)$ alternative where the best individuals are selected from both the parent and offspring individuals, since the (μ, λ)-selection is better at leaving local optima. The fitness of offspring is evaluated against the *value at risk* constraint and the objective function which we attempt to maximize:

$$f(x) = \sum_{l=N-NL+1}^{N} p_l x_{\pi(l)}^T \xi_l \tag{9}$$

where NL is the number of leaf nodes and p_l is the probability of arriving at leaf l.

4) Recombination: We use global, intermediate recombination in which the parental state for the next generation is updated by mean value calculation over the best μ of the λ offspring individuals.

5) Self-adaptation: We use the Covariance Matrix Self-Adaptation Evolution Strategy (CMSA-ES) based on a covariance learning rule [9]. The covariance matrix C is initialized with an identity matrix, updated at the end of each generation and provides main source of variations for future mutation. The mutation step size σ also evolves over generations and even each offspring individual has its own step size in the same generation. Such a self-adaptive approach is able to follow the optimum and learn efficiently.

C. Handling Transaction Costs

To pursue a more realistic model, we consider fixed costs c_f and different costs for buying c_b and selling c_s in both initialization and mutation. For each transaction (weight change), we define the total cost as follows:

$$\begin{aligned} C(x^{(t-1)}, x^t, \xi) = c_f &\sum_{m=1}^{M} \overline{\delta}(x_m^{(t-1)} - x_m^{(t)}) \\ &+ c_b \sum_{m=1}^{M} \Theta(x_m^{(t)} - x_m^{(t-1)}) |x_m^{(t-1)} - x_m^{(t)}| (\xi)_m \\ &+ c_s \sum_{m=1}^{M} \Theta(x_m^{(t)} - x_m^{(t-1)}) |x_m^{(t-1)} - x_m^{(t)}| (\xi)_m \end{aligned}$$

$$\overline{\delta}(x) = \begin{cases} 1, & \text{if } x \neq 0. \\ 0, & \text{if } x = 0. \end{cases}$$

$$\Theta(x) = \begin{cases} 1, & \text{if } x \geq 0. \\ 0, & \text{if } x < 0. \end{cases}$$

The new weight $x^{(t)}$ can be solved from (1) with iterative projection.

V. EXPERIMENTS AND DISCUSSION

We implemented our Evolution Strategy with both absence and presence of transaction costs and used Markowitz mean-variance model as the baseline[2]. We focused on a 5-asset portfolio ('MCD', 'NVDA', 'JPM', 'CVX', 'LMT') with initial capital $1,000,000. The cumulative wealth obtained by a strategy is used as the metric to measure the performance.

[1] https://github.com/charleschengxu/bolt

[2] Source code at https://github.com/charleschengxu/zeus

A. Evolution Strategy without Transaction Costs

For numerical tractability, we developed a less complicated tree with 10 levels to model the asset price change over 10 consecutive months. At each generation, 10 offspring individuals are generated and the mean value of the best 5 are used to update the next parental state.

1) Profitability of Evolution Strategy: Overall, the dynamic ES achieves higher cumulative wealth than the stationary Markowitz approach, shown in Figure 1. With optimized, evolving weights at each time step, ES generates a set of portfolio sequences with a final wealth of $1,543,834 on the average, while the Markowitz approach gives a single less profitable plan with $1,401,491.

To evaluate the robustness of strategies learned from the training period, we applied the weights on the test period that immediately follows the training period. Figure 2 shows that ES has an equally promising performance on the test period. The average final wealth from ES is $1,680,962, while the Markowitz approach only produces $1,388,312.

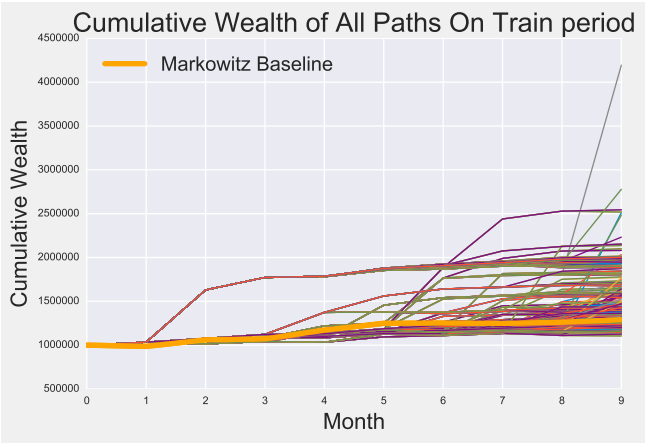

Fig. 1. Portfolio sequences generated by ES for training period

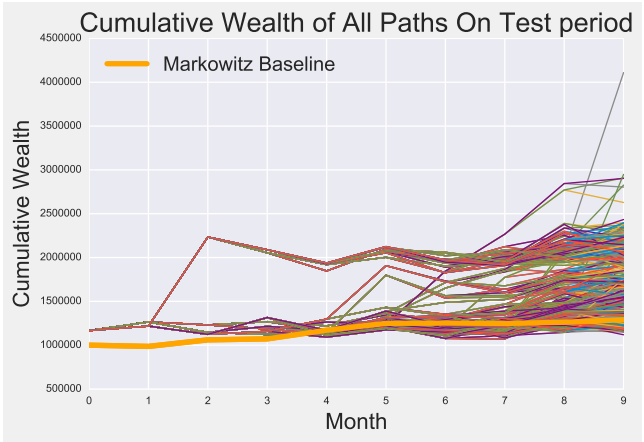

Fig. 2. Portfolio sequences generated by ES for test period

2) Path Diversity: Another observation is that with its tree structure ES is able to generate a variety of possible investment plans, shown in Figure1 and Figure 2. Some randomly sampled paths in Figure 3 reveal that such a property can be leveraged to fulfill different financial priorities. For instance, a risk-taking investor with a short-term goal would favor a path with profitable inner nodes, while a risk-averse investor with a long-term goal would stick to a path with high stability.

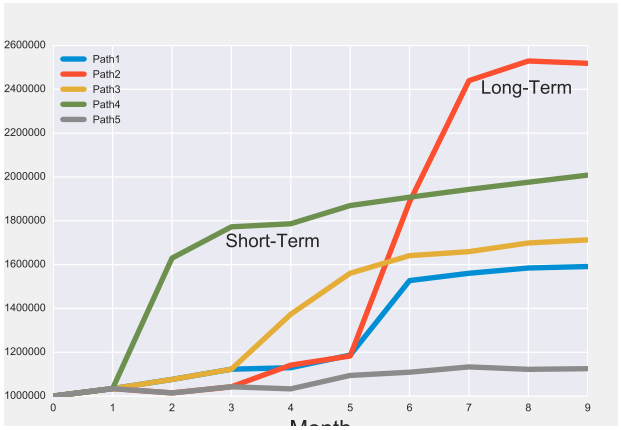

Fig. 3. Various investment strategies that fulfill different priorities

3) Smart Re-balancing: To ensure the ES algorithm yields reasonable weight adjustment along the way and understand how it works, we backtracked the complete paths leading to interested leaf nodes.

Figure 4. shows an overall increasing price pattern of all assets. Compared to Figure 5, we observe that ES re-balances the portfolio by assigning a lower weight to assets with a negative rate of change.

Figure 6 focuses on a single asset "NVDA" and confirms such interaction between price and portfolio weight. Similarly, the change in weights depends heavily on the speed of price change. If the price of an asset mildly increases, it is considered less favorable in the portfolio.

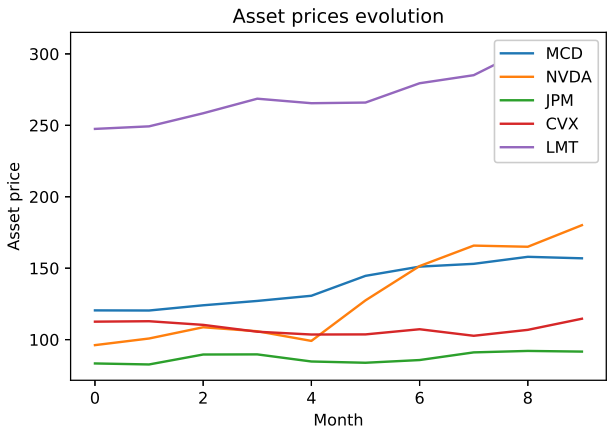

Fig. 4. Price change of the five assets over the ten months

4) Path Consistency: To measure the applicability of trading signals learned from the training period to the test

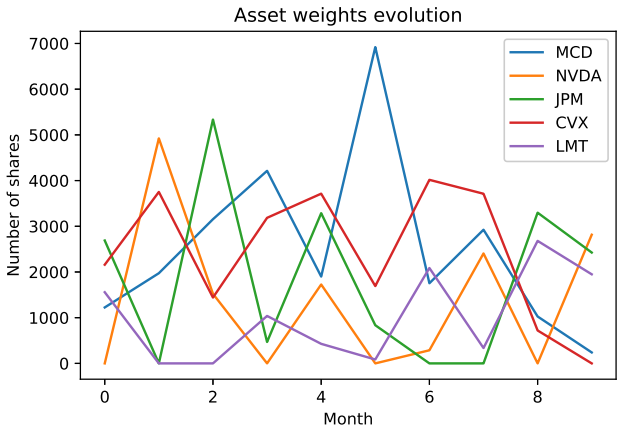

Fig. 5. Weight change of the five assets over the ten months

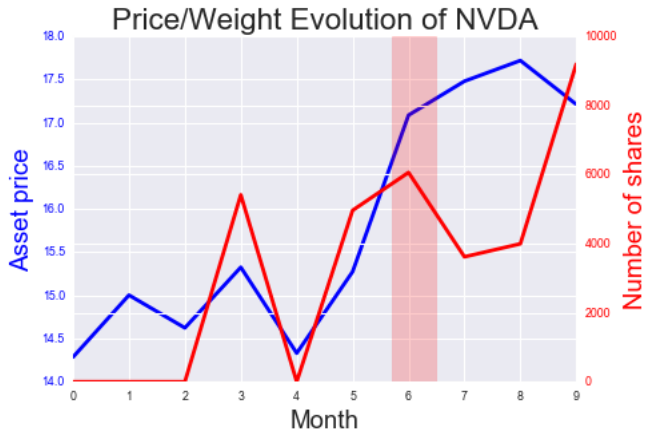

Fig. 6. Price and weight change of asset 'NVDA'

period that immediately follows, we compared the cumulative wealth achieved by the same path at each time step in both the training and test period. Figure 7 shows that a particular strategy exhibits similar performance on the two periods. The result suggests the predictive capabilities of ES: weights learned from a past period could guide transactions for a future period. A sliding window approach could also be applied to ensure the learned weights are up-to-date.

B. Evolution Strategy with Transaction Costs

With cost complexities, ES behaved more conservatively and gave a lower expected final wealth. We found out that the current algorithm which repairs the mutation direction under the cost constraint converged slowly. When it failed to find any feasible mutation direction, it simply returned the original weight in the parent node as a work-around. Thus, the portfolio changed minimally over the time. From a different perspective, frequent buy/sell actions sacrifice part of wealth and are considered less desirable in the cost model.

C. Error Analysis

Initially, we noticed that the M-th asset in the portfolio frequently had zero weight at the end of the planning

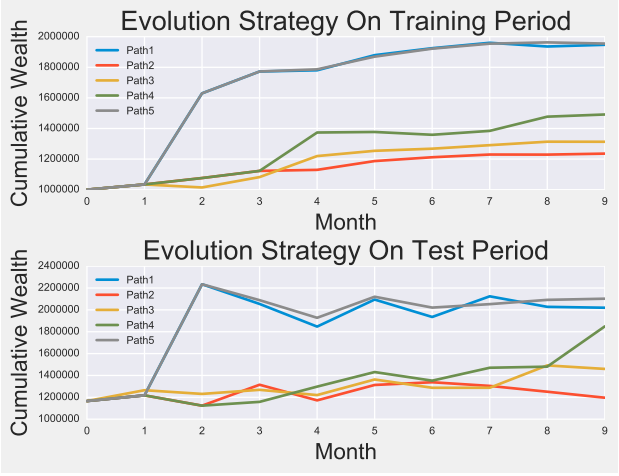

Fig. 7. Performance comparison of the same path over training/test period

horizon. It was because the projection algorithm would set the negative mutated weight to zero and only consider non-zero components in the following optimization process. Therefore, it would be biased to always fix the first $(M-1)$ components in the weight vector and determine the M-th one from (1). To resolve this, we revised the algorithm and randomly selected $(M-1)$ assets to assign weights in mutation.

VI. FUTURE WORKS

A. Tackle the Computational Bottleneck

Although in ES offspring generation was scaled up to multiple parallel workers, workloads turned out to be memory bound due to the high dimensionality of the search space. The giant covariance matrix has a size of $(M-1)N \times (M-1)N$, where M is the number of assets in the portfolio and N is the number of nodes in the scenario tree. To generate an offspring individual, the square root of the covariance matrix is calculated via spectral decomposition. This requires the solution of the eigenvalue problem and can be computationally demanding.

Therefore, we plan to explore alternative approaches that approximate eigenvalue decomposition of the covariance matrix. We may also integrate our custom, domain-specific algorithm with existing evolutionary computation frameworks such as DEAP (Distributed Evolutionary Algorithms in Python) [10] to speed up the evolution.

B. Flexible Tree Structure

Once the evolution process could proceed more efficiently, we will refine our model by considering more tree levels or proposing a different tree structure. With the current binary hierarchy, some paths seem to share some degree of similarity. Moreover, we will explore the best sliding window size (i.e. the optimal re-balancing period) by associating each level with quarterly/yearly prices besides monthly ones.

C. Better Evolution Strategies

We will also evaluate different recombination schemes that may outperform the current mean value approach in leaving local optima.

VII. CONCLUSION

In summary, the proposed Evolution Strategy performed on a scenario tree provides a promising solution to the multi-asset multi-period portfolio optimization problem where the traditional mean-variance approach only yields a stationary progress. ES allows for great flexibility by dynamically re-balancing asset weights and offering diverse investment plans. We therefore recommend this technique for making decisions for future investments and will keep working on the possible extensions of the current model.

CONTRIBUTION

Kexin collected papers on Evolution Strategy, implemented our Evolution Strategy model and gathered results. Charles automated stock prices collection from NYSE and Nasdaq, implemented Markowitz mean-variance optimization as the baseline and gathered results. We coauthored this paper and contributed equally.

REFERENCES

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

[2] W. F. Sharpe, "The sharpe ratio," *The journal of portfolio management*, vol. 21, no. 1, pp. 49–58, 1994.

[3] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical finance*, vol. 9, no. 3, pp. 203–228, 1999.

[4] A. A. Gaivoronski and G. Pflug, "Value-at-risk in portfolio optimization: properties and computational approach," *The Journal of Risk*, vol. 7, no. 2, p. 1, 2004.

[5] C. Acerbi and D. Tasche, "Expected shortfall: a natural coherent alternative to value at risk," *Economic notes*, vol. 31, no. 2, pp. 379–388, 2002.

[6] G. W. Schwert and P. J. Seguin, "Heteroskedasticity in stock returns," *The Journal of Finance*, vol. 45, no. 4, pp. 1129–1155, 1990.

[7] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution Strategies as a Scalable Alternative to Reinforcement Learning," *ArXiv e-prints*, 2017.

[8] T. B. Hans-Georg Beyer, Steffen Finck, "Evolution on trees: On the design of an evolution strategy for scenario-based multi-period portfolio optimization under transaction costs," *Swarm and Evolutionary Computation*, vol. 17, pp. 74–87, 2014.

[9] H. georg Beyer and B. Sendhoff, "Covariance matrix adaptation revisited: the CMSA evolution strategy," *Parallel Problem Solving from Nature*, vol. 10, pp. 123–132, 2008.

[10] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.