Portfolio Optimization with Monte Carlo Simulations: Practical Allocation Model

By: Mateo Florsheim

Table of Contents:

1. Summary - This project develops a quantitative portfolio construction framework designed to improve the practical usability of mean-variance optimization under noisy and unstable inputs. The model combines constrained portfolio optimization with correlation denoising, implementable allocation logic, and regime-aware Monte Carlo simulation to produce portfolios that are statistically grounded and operationally realistic.

The model estimates expected returns and covariance from historical market data, applies Marchenko-Pastur filtering to reduce correlation noise, and dives into a long-only optimization problem across equities, cash, and T-Bills. It then converts continuous target weights into discrete portfolio allocations and evaluates forward outcomes using a Hidden Markov Model (HMM) based Monte Carlo engine allowing return dynamics to vary across market regimes.

Out-of-sample testing shows that the portfolio construction framework could generate positive realized outcomes across many simulations, but that overall performance was sensitive to expected-return estimation. Baseline estimators systematically overstated expected performance, while more structured approaches such as Black-Litterman improved calibration and produced a more credible balance between forecasted and realized results.

The central conclusion of the project is that portfolio optimization is not limited by the optimization route itself, but by the quality and stability of its inputs, especially expected returns. As a result, the model is best understood not as a return maximization engine, but as a research platform for improving portfolio construction under estimation risk.

2. Motivation - Traditional mean-variance optimization remains a starting point for portfolio construction, but its performance is limited by estimation error rather than by underlying optimization. In practice, expected returns are difficult to estimate, sample correlation matrices are noisy, and unconstrained solutions often produce allocations that are unstable or difficult to implement.

These issues are relevant when the number of assets is large relative to the amount of available history. In that setting, optimization can overreact to small changes in estimated returns and correlations, leading to concentrated or unintuitive portfolios that do not hold up well out of sample.

This project addresses those practical weaknesses directly. The framework combines correlation denoising, constrained portfolio construction, and regime-aware simulation to produce a process that is more robust to noisy inputs and more aligned with real-world allocation decisions.

Rather than treating portfolio optimization as a forecasting exercise, the report evaluates it as a portfolio construction problem under uncertainty. That framing is important because the objective is not to generate extreme return forecasts, but to build a more credible and implementable allocation process.

3. Methodology

3.1 Model Inputs - This model operates on a structured set of user inputs defining the investment universe, portfolio constrains, and simulation parameters.

At the asset level, the portfolio manager specifies a set of equities along with the maximum allowable position sizes per ticker. At the portfolio level, the portfolio manager defines the total capital, a historical lookback window, and a risk-aversion parameter that governs the tradeoff between return and risk. The risk aversion parameter is measured using the below equation:

$$U(w) = \mu^T w - \frac{1}{2}\lambda w^T \Sigma w$$

The segments $\mu^T w$ = expected return, $w^T \Sigma w$ = portfolio variance (risk), and λ = risk aversion parameter. Think of λ, or risk aversion, as how much return one is willing to give up reducing risk. As λ decreases risk penalty weakens, the optimizer allocates more high-returning stocks, accepts higher volatility, and may concentrate in fewer names. A higher risk aversion value is conservative investor prioritizing stability, penalizing volatility, shifting toward more cash and T-bills. In the model, it represents a portfolio variance penalty increasingly, reduces equity exposure, increases cash, and diversifies more as λ increases.

In addition to defining the investment universe and core portfolio parameters, the model incorporates several inputs that govern allocation constraints and defensive positioning.

At the portfolio level, the portfolio manager specifies minimum and maximum cash allocations, ensuring that the portfolio maintains a required level of liquidity while preventing excessive idle capital. A cash yield parameter is also including this allows uninvested capital to earn a return rather than being treated as a zero-yield residual. All inputs that represent percentages will be put in a whole number not decimal form.

The model further incorporates a Treasury-Bill (T-Bill) sleeve, which serves as a low-risk alternative to holding excess capital in cash. While cash provides liquidity, T-Bills allow the portfolio to earn a return on defensive capital. The model can either use a user-defined T-Bill yield or automatically retrieve a market-based rate via yfinance, when available. A maximum allocation constraint is applied to this sleeve to prevent over-allocation to low-risk assets while allowing the optimizer to reduce exposure to equities when risk-adjusted returns are unattractive.

In this model, the defensive portion of the portfolio is treated as a combined allocation across cash and T-Bills rather than as separate independent decisions. This allows the optimizer to determine the most efficient mix of liquidity and low-risk yield while maintaining required capital preservation.

The model also includes flexibility in how asset allocation constraints are handled. In manual mode, the portfolio manager directly specifies maximum weights for each asset. In automatic mode, the model generates allocation caps based on asset characteristics such as volatility and correlation. Assets with lower volatility and stronger diversification properties are allowed slightly higher allocations, while more volatile or highly corelated assets receive tighter limits. This creates a balance between diversification and return-seeking behavior without requiring the portfolio manager to manually tune each constraint.

Additional inputs define the statistical and simulation environment. The lookback window determines the amount of historical data used to estimate returns and risk. A longer lookback period generally produces more stable estimates, longer predicted correlation, while shorter windows allow the model to respond more quickly to recent market conditions. This is useful for portfolio managers looking to rebalance more often.

The portfolio manager also defines parameters for the Monte Carlo simulation, including the number of simulation paths and the investment horizon. These inputs control the depth and time frame of the forward-looking analysis and directly impact the stability and interpretability of the simulated outcome distribution.

Finally, the model support optional portfolio targets such as a minimum expected return or a maximum allowable volatility. These constraints allow the portfolio manager to guide the optimization toward specific performance or risk objectives, ensuring that the resulting portfolio aligns with defined investment goals.

Taken together, these inputs define the feasible investment universe and impose realistic constraints on the optimization process. Rather than producing purely theoretical outputs, the model translates inputted preferences and market data into portfolios that are both statistically grounded and practically implementable.

3.2 Return and Risk Estimation - The model estimates expected return and risk directly from historical adjusted close price data. This approach allows the optimizer to rely on observable market information rather than subjective assumptions.
Daily log returns are computed for each asset using the following expression:

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

P_t represents the adjusted closing price at time t. Log returns are used because they are time-additive and provide a consistent framework for measuring returns over multiple periods.

For each asset, expected return is estimated at the average of historical daily log returns, annualized using the standard approximation of trading days per year:

$$\mu_i = \text{mean}(r_i) \times 252$$

This produces an annualized expected return for each stock based on the selected lookback window.

Risk is measured using the covariance structure of asset returns. The model computes the covariance matrix from the same return series and annualizes it as shown:

$$\Sigma = cov(r) \times 252$$

The diagonal elements of the covariance matrix represent individual asset variances, while the off-diagonal elements capture the relationships between assets. These relationships are critical for measuring diversification effects within the portfolio universe.

From the covariance matrix, the model derives annualized volatility for each asset:

$$\sigma_i = \sqrt{\Sigma_{ii}}$$

Volatility serves as the primary measure of standalone asset risk, while the full covariance matrix determines how risk behaves at the portfolio level.

In addition to covariance, the model computes the correlation matrix to better understand the degree of corelating movement between assets. Correlation plays a central role in portfolio construction, as it determines how effectively risk can be reduced through diversification.

While these estimates are based on historical data, they provide a practical and widely used foundation for modeling expected return and risk. These outputs serve as the core inputs for the optimization process, which is further refined through correlation denoising in the next stage of the model. Factor-based return models provide a more structure approach to expected-return estimation and represent a natural extension beyond the estimators tested in this model.

3.3 Correlation Denoising (Marchenko-Pastur) - One the of main limitations of traditional mean-variance optimization is its sensitivity to estimation error, particularly in the correlation matrix. When the number of assets is large relative to the number of observations, empirical correlations tend to contain a significant amount of noise. This noise can lead to unstable or unrealistic portfolio allocations, as the optimizer may overfit relationships that are not truly persistent.

To address this issue, the model applies a denoising procedure based on the Marchenko-Pastur distribution from random matrix theory, this approach provides a theoretical framework for distinguishing meaningful structure from random noise in large correlation matrices.

Let *T* represent the number of time observations and *N* represent the number of assets. Defined as:

$$Q = \frac{T}{N}$$

Under the assumption of random, uncorrelated data, the eigenvalues - a value were multiplying a square matrix by a non-zero vector scales the vector without rotating it defined as $Av = \lambda v$ - of the correlation matrix are expected to fall within a specific range. The upper bound of this range is given by:

$$\lambda_{+} = \left(1 + \frac{1}{\sqrt{Q}}\right)^2$$

Eigenvalues that fall within this range are dominated by noise, while eigenvalues above this threshold are more likely to represent meaningful market structure.

The model implements the following steps listed:

1. Compute the empirical correlation matrix from asset returns.
2. Perform eigenvalue decomposition to obtain eigenvalues and eigenvectors
3. Compare each eigenvalue to the Marchenko-Pastur upper bound.
4. Treat eigenvalues within the noise range as unstable and replace them with a smoothed estimate.
5. Preserve eigenvalues that exceed the threshold, as they capture significant relationships.
6. Reconstruct a cleaned correlation matrix using the adjusted eigenvalues.

Once the denoised correlation matrix is obtained, the model rebuilds the covariance matrix using the estimated asset volatilities:

$$\Sigma_{clean} = D \times Corr_{clean} \times D$$

Where D is the diagonal matrix of asset volatilities.

This process reduces the impact of spurious correlations while preserving the dominate structure in the data. As a result, the optimization becomes more stable and less sensitive to small changes in input data.

The denoising procedure is based on Marchenko-Pastur Distribution, which separates signal from noise in empirical correlation matrices. By incorporating Marchenko-Pastur filtering, the model improves robustness and reliability of portfolio construction. This step is especially important in higher dimensional settings, where noisy estimates can otherwise lead to extreme or poorly diversified allocations. While this model uses spectral denoising, alternative approaches such as shrinkage-based covariance estimation provide a complementary method for improving stability and represent a potential extension.

3.4 Portfolio Optimization Framework - The portfolio optimization model is based on Modern Portfolio Theory, originally developed by Harry Markowitz. The model maximizes a risk adjusted utility faction that trades off expected return and portfolio variance. The original version of the model uses a constrained mean-variance optimization framework to determine portfolio allocations across equities, cash, and Treasury bills. This baseline implementation follows the standard Markowitz formulation and serves as the foundation for the initial OOS (Out of sample) back testing results. The optimization problem is defined by the objective:

$$U(w) = \mu^T w - \frac{1}{2}\lambda w^T \Sigma w$$

Where w represents the portfolio weights, μ is the vector of expected returns estimated from historical data, and Σw is the covariance matrix derived from asset returns. The first term represents expected portfolio return, while the second term penalizes total portfolio variance.

In the baseline model, the risk-aversion parameter λ directly controls the tradeoff between return and risk. Lower values of λ lead to more aggressive portfolios that prioritize expected return, while higher values place greater emphasis on risk reduction and diversification. The portfolio is constructed across 3 components: equities, cash, and Treasury bills form a defensive allocation. Cash provides liquidity, while Treasury bills offer a low-risk yield alternative to idle capital.

The optimization is subject to several constraints. All portfolio weights must sum to 1, ensuring that all capital is allocated. The model enforces a long-only constant, preventing short positions. Each stock is also subject to a maximum allocation cap, which is either specified directly by the portfolio manager or applied uniformly across the portfolio.

The defensive sleeve is controlled through minimum and maximum allocation constraints of the combined weight of cash and T-Bills. This ensures that the portfolio maintains a required level of capital preservation while still allowing flexibility in how that defensive allocation is distributed.

The model also supports optional constraints such as a minimum expected return or maximum allowable volatility. These constraints allow the optimizer to target specific portfolio characteristics, though they can reduce feasibility if the required thresholds are too restrictive.

The optimization is solved numerically using constrained nonlinear methods. The model searches over feasible portfolio weights to identify the allocation that maximizes the objective function while satisfying all constraints.

This baseline model produces a set of optimal portfolio weights based on historical return estimates and covariance structure. However, because it relies heavily on raw statistical estimates, it is sensitive to estimation error in both expected returns and correlations.

This sensitivity is reflected in the original OOS back testing results. While the model frequently produced portfolios with positive realized returns, the expected-return forecasts were often overly optimistic. As a result, many portfolios were profitable in absolute terms but failed to meet the model's predicted return threshold, leading to relatively low formal prediction accuracy despite strong profitability rates.

3.5 Portfolio Construction & Implementation - The optimization process produces a vector of continuous portfolio weights, but these weights are not directly implementable in practice. Real portfolios must be expressed in discrete units, such as whole shares, and must account for capital constraints and residual cash balances. This section describes how the model translates theoretical allocations into an executable portfolio.

The first step is to convert optimized weights into dollar allocations. Given total portfolio capital, each asset's target dollar position is calculated by multiplying its weight by total capital. This produces a continuous target allocation across all assets, including equities, cash, and T-Bills.

For equities, these dollar allocations are then converted into discrete share counts. Since fractional shares are not assumed, each position is rounded down to the nearest whole share. This rounding process introduces small deviations for the original optimized weights, as the exact target allocation may not be perfectly achievable.

After rounding, the model computes the remaining unallocated capital. This residual capital arises from the difference between the continuous allocation and discrete share implementation. Rather than leaving this capital unused, the model reallocates it in a structured way.

First, the required defensive sleeve is satisfied. The model ensures that the combined allocation to cash and T-Bills meets the minimum constraint specified by the portfolio manager. Within this sleeve, Treasury bills are used to generate yield on defensive capital, while cash is retained for liquidity purposes.

Any remaining capital beyond the minimum defensive requirement is then distributed across equity positions. The model prioritizes assets where the gap between the target allocation and the implemented allocation is largest. This helps reduce tracking error between the optimized portfolio and the final implemented version.

The final output of this model is a fully specified portfolio including:

1. Exact share counts for each stock
2. Dollar allocation per asset
3. Total allocation to cash
4. Total allocation to T-Bills

This implementation step is important as it bridges the gap between theoretical optimization and real-world execution. While small deviations from the optimal weights are unavoidable due to discrete constraints, the model ensures that these deviations are minimized and handled systematically. By incorporating portfolio construction directly into the framework, the model produces results that are not only optimal in theory but also verified on real world information.

3.6 Regime Switching Monte Carlo Simulation (HMM) - The Monte Carlo simulation produces a distribution of possible portfolio outcomes over the selected investment horizon. This distribution is generated using regime-switching return dynamics and reflects both return uncertainty and changes in market conditions.

For each optimized portfolio, the model runs a specified number of simulation paths (5k+). Each path represents a possible evolution of portfolio value based on stochastic returns and regime transitions. The result is a full distribution of terminal portfolio values rather than a single estimate.

The model evaluates forward portfolio performance using the regime-switching Monte Carlo simulation driven by a Hidden Markov Model (HMM). It allows return distributions to vary over time instead of assuming a single stationary process.

Regime Identification

The model constructs a market factor using the cross-sectional average of asset returns. This time series is used to fit a Gaussian Hidden Markov Model, which assumes returns are generated by a finite set of latent regimes. Each regime is defined by:

1. Mean return
2. Variance
3. Transition Probabilities

The HMM estimates these parameters from historical data and infers the current regime probabilities.

Regime-Specific Return Estimation

After regime identification, the model partitions historical returns by regime. For each regime k, it computes μ^k, Σ^k where μ^k is the regime-specific expected return vector and Σ^k is the regime-specific covariance matrix. This allows the model to represent different market conditions, such as low-volatility and high-volatility environments, each with distinct return characteristics.

Monte Carlo Simulation Process

The simulation generates future portfolio paths using regime transitions stochastic return draws. For each simulation path:

1. Initialize the starting regime using current regime probabilities.
2. At each time step:
    a. Sample the next regime using the Markov transition matrix
    b. Draw asset returns from a multivariable normal distribution:
        i. $r_t \sim N(\mu^k, \Sigma^k)$ where k is the current regime.
3. Update portfolio value.
4. Repeat over the full simulation horizon.

This process is repeated across multiple simulation paths to generate a distribution of outcomes.

Portfolio Evolution

At each step:

1. Equity positions evolve based on simulated returns
2. Cash accrues its specified yield
3. T-Bills accrue their assigned yield

Total portfolio value is updated sequentially across the simulation horizon

Simulation Outputs

After all paths are generated, the model summarized terminal outcomes including expected terminal value, median terminal value, percentile ranges, expected returns over the horizon, and probability of loss. These metrics describe the distribution of potential outcomes rather thana single estimate.

Role in the Framework

The optimization step produces a single allocation based on historical estimates. The Monte Carlo layer evaluates that allocation under varying market conditions. By incorporating regime switching, the model avoids assuming constant volatility and correlation. Instead, it reflects changing market environments, producing a more realistic assessment of portfolio risk and return.

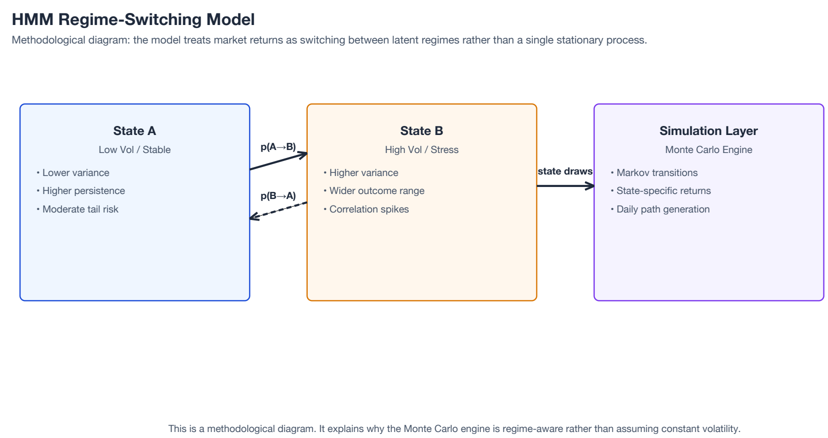


Figure 1

Hidden Markov Model regime identification illustrating transitions between market states used in simulation

Figure 1 represents the two latent states (State A = low vol; State B = high vol) feed state-specific return draws into the Monte Carlo engine. Arrows show the transition probabilities, and the footer reiterates this is a conceptual illustration supporting the regime-aware simulation.

The set underlying this framework includes both methodologies that are directly implemented in the model and those informing its development path. The current implementation revolves around the Modern Portfolio Theory, with covariance denoising based on the Marchenko-Pastur Distribution and regime dynamics modeled using a Hidden Markov Model. Expected return estimation incorporates both Bayes-Stein Estimation and Black-Litterman Model approaches. Additional references, including covariance shrinkage methods, factor-based return models, volatility managed portfolios, and Bayesian allocation, are included to support future extensions of the model.

4. System Architecture - Taking in all the previous section's information, System Architecture simplifies the different layers of the model. The system is organized as a modular pipeline that processes user inputs, constructs the portfolio, and evaluates forward performance. Each component is separated by function to ensure clarity, flexibility, and ease of modification.

Data Layer - The system begins by retrieving historical adjusted-close price data for all user-specified tickers. This data is used to compute return series, volatility, covariance, and correlation matrices. Data preprocessing includes aligning time series, handling missing values, and converting prices into daily log returns.

Estimation layer - From the processed return data, the model estimates expected returns and risk metrics. This includes:

1. Annualized expected return vector
2. Covariance Matrix
3. Correlation Matrix

The correlation matrix is then denoised using the Marchenko-Pastur procedure to produce a cleaned covariance structure. These estimates form the core inputs for the optimization stage

Optimization Layer - The optimization layer solves the constrained mean-variance problem using the estimated inputs. It determines portfolio weights across equities, cash, and T-Bills while enforcing all constraints, including:

1. Long-Only constraint
2. Maximum allocation per asset
3. Defensive sleeve constraints
4. Optional return and volatility targets

The output of this layer is a vector of optimal portfolio weights.

Implementation Layer - The implementation layer converts continuous weights into an executable portfolio. Target weights are translated into dollar allocations and then into discrete share counts. Residual capital is allocated across cash and T-Bills, and any remaining funds are redistributed to minimize deviation from target allocations.

Simulation Layer - The simulation layer evaluates forward performance using the regime-switching Monte Carlo framework. A Hidden Markov Model is fitted to market data, and regime-dependent return distributions are used to simulate portfolio evolution over time.

This layer produces a distribution of possible outcomes and calculates risk metrics such as probability of loss and percentile ranges.

Output Layer - The final output combines optimization results and simulation metrics into a structured report. The system provides:

1. portfolio allocations and share counts
2. expected return and volatility
3. Monte Carlo distribution statistics
4. probability of loss
5. portfolio risk score and classification

This architecture separates data processing, estimation, optimization, and simulation into distinct components. This design improves transparency, allows for targeted improvements, and supports future extensions without restructuring the entire system.

5. Empirical Results

5.1 Back testing Design and Evaluation Methodology - Results are evaluated using a point-in-time out-of-sample (OOS) backtesting framework designed to separate portfolio formation from forward performance measurement. The backtests were conducted on randomly generated 10-stock portfolios drawn from a broader universe of liquid and volatile equities listed on the NYSE and NASDAQ as of January 1st, 2024.

For each simulation, a formation date is selected, and model inputs are estimated using only information available at that date. In this case, the single-date estimator comparison (Sections 5.2-5.5) used a 1-year lookback, starting January 1st, 2024. The 1-year historical lookback window is used to estimate expected returns and covariance, after which the portfolio is constructed subject to the model's allocation constraints and held over a one-year forward period with the rational that the expected return is more closely related to the current market state/regime and avoids noise.

For each completed run, the model records expected return, realized return, Monte Carlo expected return, and terminal portfolio value. Prediction accuracy is defined as the proportion of simulations in which realized return met or exceeded the model's expected return forecast and is therefore intended to measure forecast calibration rather than economic profitability.

A total of 1,000 simulations were requested, of which 994 completed successfully and 6 failed. This framework is intended to evaluate how different return estimators affect OOS portfolio behavior under a consistent portfolio construction process, rather than to present a production-ready live trading result.

5.2 Baseline Model Results - The baseline model uses the historical mean return estimator. Across 994 completed simulations, it produced a prediction accuracy of 23.44%, a median expected return of 19.31%, and a median realized return of 8.84%.

These results indicate that the baseline framework systematically overestimated expected returns. While many portfolios generated positive realized outcomes, they frequently failed to meet the model's forecasted return threshold, implying that the optimization process itself was functional, but the return estimates were not sufficiently reliable.

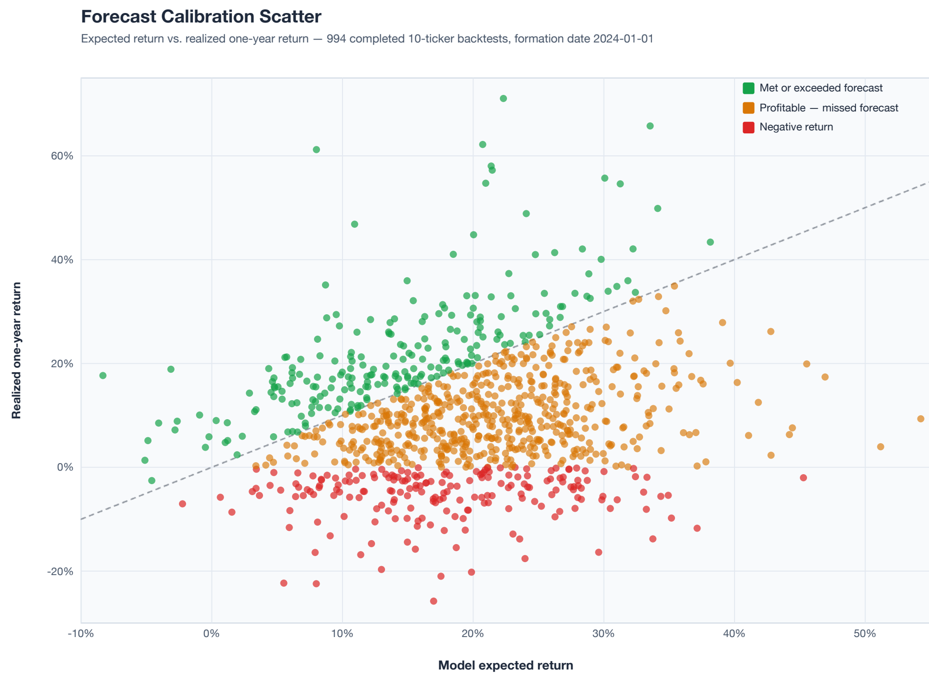


Figure 2

Figure 2. Forecast calibration scatter comparing expected and realized returns across portfolio simulations. The deviation from the diagonal reflects systematic overestimation.

Figure 2 compares the model's expected return (x-axis) to the realized one-year return (y-axis) for every 10-ticker backtest. Green dots beat their forecast, amber dots finished positive but below expectation, and red dots lost money. The diagonal line highlights that the dense amber cloud is where the model was too optimistic even though most portfolios stayed profitable. Additionally, for the 1-year test the HMM is fitted on an in-sample window (which includes bull-market data), and the regime specific return parameters inherit that optimism. The Monte Carlo engine then compounds the inflated regime means which produces exponentially greater returns. This is a methodological limitation as previous regimes can sway returns.

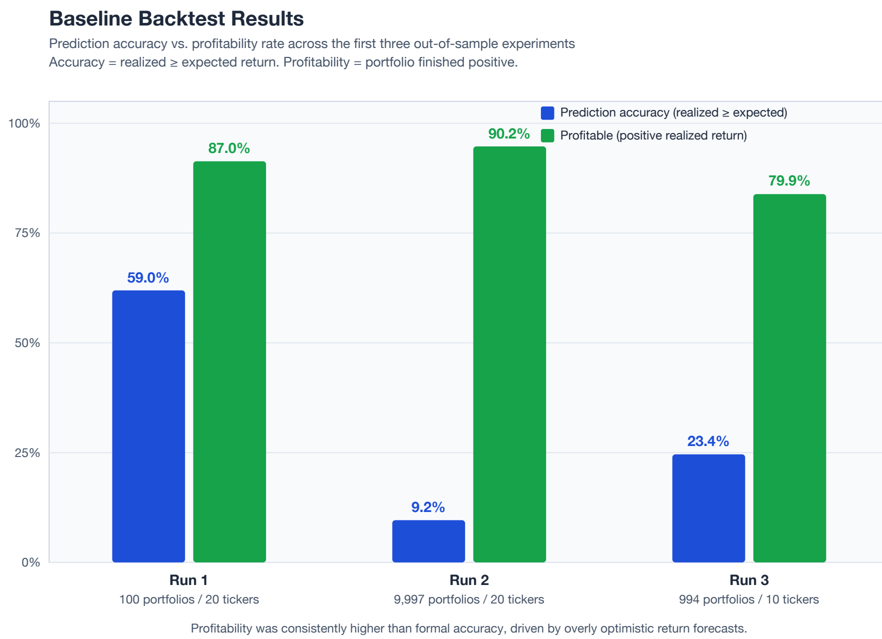


Figure 3

Figure 3. Baseline back test results summarizing portfolio performance across simulations.

Figure 3 shows baseline accuracy vs. profitability bars for the first three original experiments (100 portfolios of 20 tickers, 9,997 portfolios of 20 tickers, and 994 portfolios of 10 tickers). Each pair shows how often the model's forecast hit the mark versus just making money; profitability is materially higher than formal accuracy, motivating the upgrade work.

5.3 Updated Model Results (Return Estimator Comparison) - The historical mean estimator completed 994 simulations with 6 failures and produced an accuracy of 23.44%, a median expected return of 19.31%, and a mean realized return of 8.84%. The Bayes-Stein estimator completed 994 simulations with 6 failures and produced an accuracy of 23.24%, a median expected return of 19.30%, and a median realized return of 8.84%, suggesting limited to no improvement from shrinkage in this setting.

The market-factor estimator completed 994 simulations with 6 failures and produced an accuracy of 76.76%, a median expected return of 1.04%, and a median realized return of 8.49%. Its higher formal accuracy appears to be driven primarily by a much lower expected-return hurdle rather than a meaningfully stronger realized performance outcome.

The Black-Litterman estimator completed 994 simulations with 6 failures and produced an accuracy of 46.48%, a median expected return of 10.14%, a median realized return of 8.83%. Relative to the other approaches tested, this estimator reduced return bias and produced a more balanced alignment between expected and realized performance.

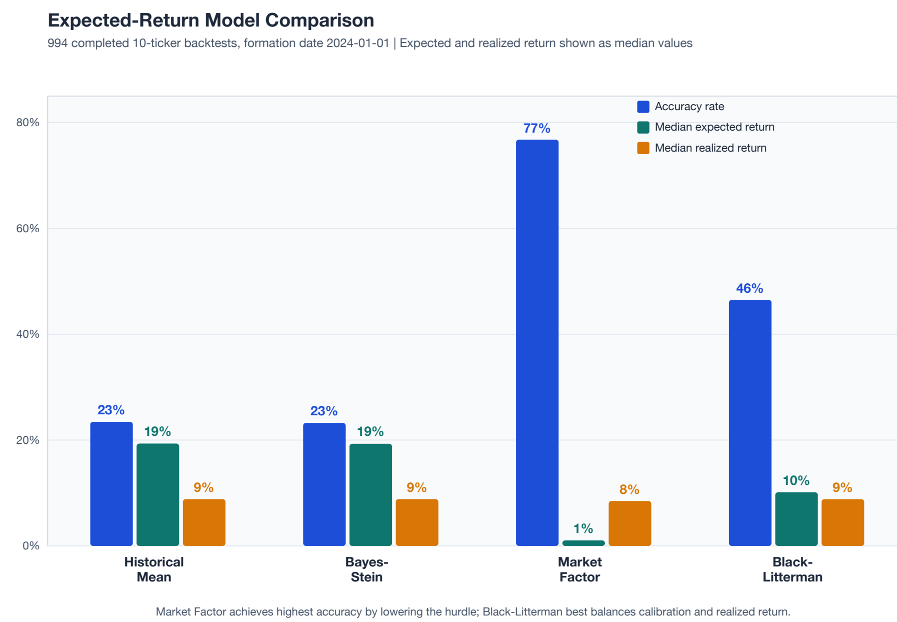


Figure 4

Comparison of prediction accuracy and expected-return calibration across return estimators.

Expected-return estimator comparison across 994 completed 10-ticker back tests. The three metrics are forecast accuracy (green), mean forecast (blue), and mean realized return (orange). The y-axis matches the actual percentages so you can see that Market Factor sacrificed upside (low forecast) for the highest accuracy, while Black-Litterman kept forecasts and realizations closer.

5.4 Comparison of Results - The results show a clear relationship between the aggressiveness of expected-return forecasts and measured prediction accuracy. Estimators that generated higher expected returns tended to produce lower accuracy because they systematically set a more difficult hurdle for realized returns to exceed.

Conversely, estimators with much lower expected-return forecasts produced higher formal accuracy, though that improvement was partly mechanical rather than purely economic. The most useful estimators were those that improved calibration while still preserving economically meaningful expected-return levels.

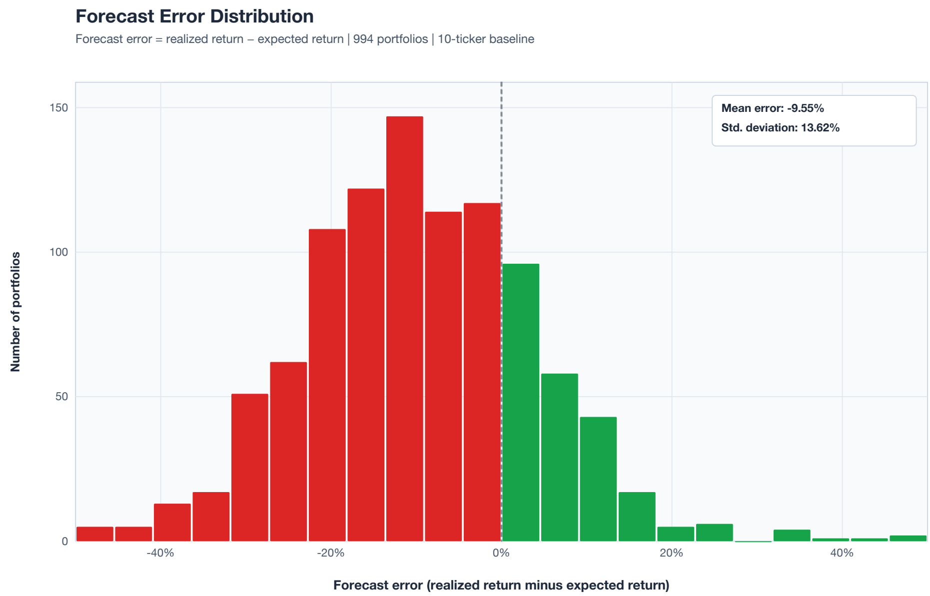


Figure 5

Distribution of forecast errors (realized minus expected returns), showing systematic bias and dispersion.

Figure 5 represents a Histogram of forecast error (realized minus expected return) for the 10-ticker baseline experiment. The mean error (~ -9.55%) and standard deviation (~ 13.62%) in the upper-right text box emphasize that the optimizer was systematically too bullish even when most portfolios finished positive.

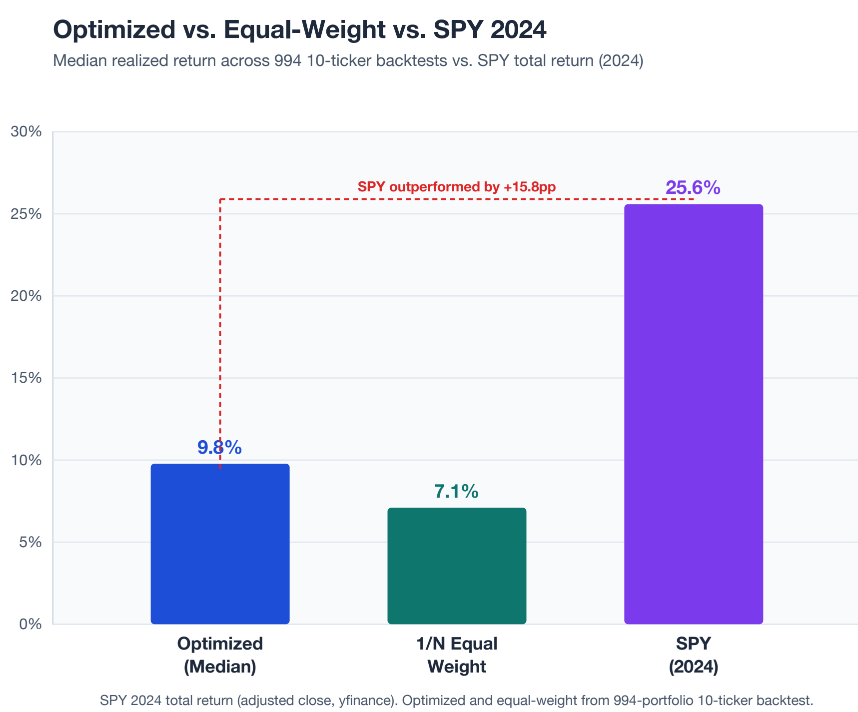


Figure 6

Portfolio Performance Comparison (Optimized vs Equal-Weight vs Market Benchmark)

Figure 6 compares the cumulative performance of the optimized portfolio against a naïve equal-weight (1/N – each ticker has the same portfolio allocation) portfolio and a market benchmark (SPY - 2024). The comparison provides a baseline for evaluating whether the optimization framework delivers incremental value over simple diversification and passive market exposure. My example of 10 tickers includes APP, AUPH, BRZE, BTAI, GLPG, IMAX, IRWD, MAXN, MLKN, and SE as a random selected 10 ticker subset.

While the tested portfolios underperformed SPY over the 2024 evaluation period, the optimizer still achieved its intended function. The optimized allocations outperformed a naïve equal weight baseline across the same ticker universe. This suggests that the optimization process adds value at the portfolio construction level, even when the underlying stock selection does not keep pace with the broad market.

5.5 Interpretation of Results - The results indicate that expected-return estimation is the primary limitation of the model. Across the estimators tested, realized returns remained relatively stable, while differences in formal forecast accuracy were driven mainly by how optimistic each model's expected-return forecast was.

The historical mean estimator produced systematically inflated return expectations, which reduced calibration despite frequently positive realized outcomes. Bayes-Stein shrinkage offered only limited improvement in this specific testing setup, while the market-factor estimator improved formal accuracy largely by lowering the forecast hurdle to a level that was easier to exceed.

Black-Litterman produced the most balanced result among the estimators tested. It reduced return bias and brought expected and realized outcomes closer together without collapsing expected returns to economically trivial levels.

The broader implication is that improving portfolio construction depends more on strengthening the quality and stability of return inputs than on making the optimizer itself more complex. Within this framework, estimation quality mattered more than optimization complexity.

These single-date results are extended in Section 5.6 with a 4-year rolling window test across 1,000 portfolios per year, providing a more robust picture of the model calibration across different market regimes.

5.6 Multi-Year and Rebalancing Tests – To evaluate model robustness beyond only 2024, 2 large-scale rolling window backtests were conducted using 1000 unique 10-ticker portfolios per formation year, drawn from a 691-ticker universe of liquid NYSE/NASDAQ equities.

Below is a rolling window (No rebalance) table where each portfolio was fitted on 3 years of in sample data and evaluated on the following 1-year OOS period, with formation years being 2021-2024, with 2022 bear market outcomes. (Figure 9)

Formation Year	**OOS Accuracy**	**Median Realized Return**	**Portfolios**
2021	38.7%	+20.4%	953
2022	4.5%	-11.8%	932
2023	90.2%	+18.1%	951
2024	63.4%	+7.8%	988

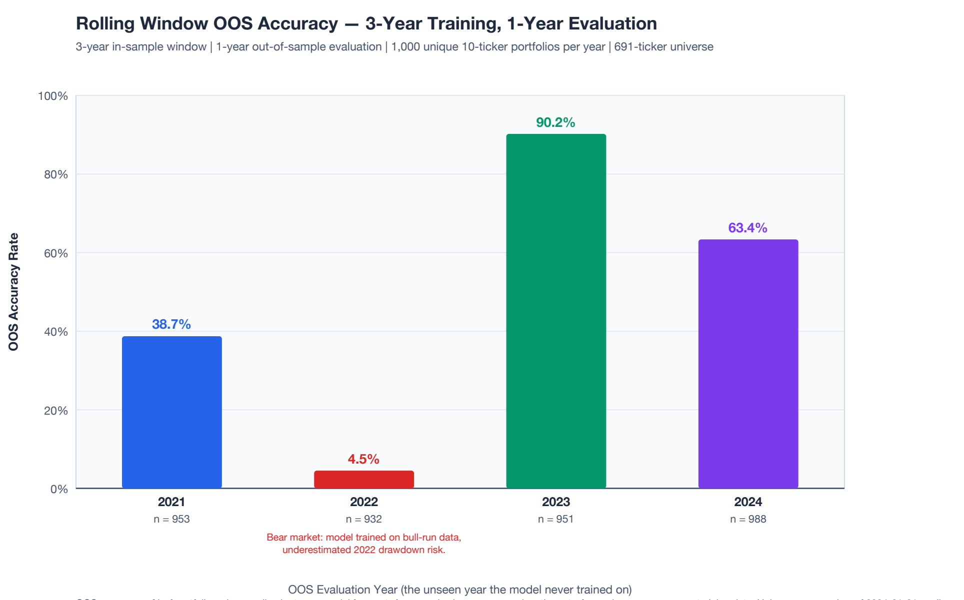


Figure 7

Figure 9 illustrates how regime shifts directly impair forecast calibrations. Training on the 2019-2021 bull market left the model overconfident entering 2022. It projected strong positive returns that never materialized, producing just 4.5% OOS accuracy and a median realized return of -11.8%. the 2022 bear market then had the opposite effect - the model absorbed negative return data and projected conservative, near zero expected returns for 2023. Because the 2023 market recovered sharply (18.1% median realized), realized returns exceeded those pessimistic forecasts in 90.2% of portfolios. By 2024, with more balanced training data, accuracy settled at 63.4%, closer to what a well-calibrated model should produce in a normal market. No rebalancing brought median betas between 1.176 and 1.275.

Quarterly Rebalancing: The same universe was re-run with quarterly re-optimization at January, April, July, and October, with 4 quarterly realized returns compounded into an annual figure (Figures 10 & 11).

Formation Year	Median Realized Return	Profitable Portfolios
2021	+15.3%	85.5%
2022	+2.4%	56.3%
2023	+14.4%	87.9%
2024	+13.0%	79.5%

Figure 8

Figure 9

The rebalancing advantage is most visible in 2022: buy-and-hold lost a median of -11.8% while quarterly rebalancing returning +2.4%, keeping 56% of portfolios profitable versus only 9.5%. This suggests quarterly re-optimization allows the model to defensively tilt toward cash and T-bills as drawdown data accumulates. in normal years (2023-2024), rebalancing delivered similar upside while maintaining high profitability rates. Rebalancing brought median betas between 1.170 and 1.286.

6. Portfolio Interpretation

The model should be interpreted as a portfolio construction and risk-framing tool rather than as a pure return-prediction engine. Its outputs are most useful when evaluated through both allocation characteristics and distribution-based risk metrics, rather than through expected return alone.

In this model, risk is assessed through portfolio volatility, simulated probability of loss, and the dispersion of terminal outcomes across Monte Carlo paths. These measures provide a more complete view of portfolio quality by distinguishing between portfolios that may have similar expected returns but meaningfully different downside exposure.

Cash and Treasury-bill allocations also play a structural role in portfolio interpretation. They are included not as residual allocations, but as defensive components that can reduce downside sensitivity and stabilize portfolio outcomes when risk-adjusted equity opportunities are less attractive.

The practical value of the framework lies in its ability to compare portfolios on robustness as well as return potential. In that sense, the model is best viewed as a disciplined allocation process under uncertainty rather than as a standalone forecasting mechanism.

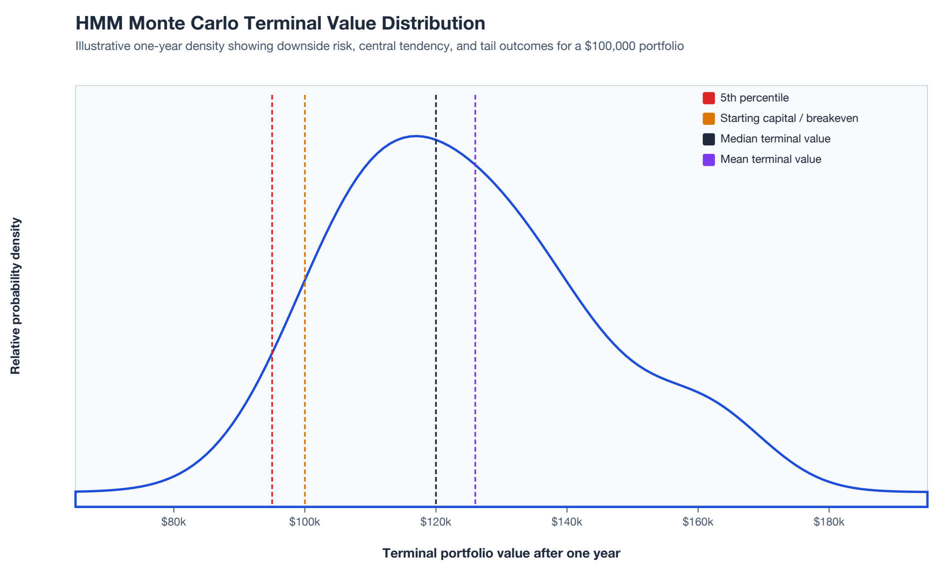


Figure 10

Monte Carlo distribution of simulated portfolio outcomes, illustrating dispersion and downside risk.

Figure 7 represents conceptual terminal-value distribution from the HMM Monte Carlo engine. The peak shape shows the central tendencies, while the colored vertical lines highlight the 5th percentile, starting capital/breakeven, median, and mean terminal values. The legend sits in the upper-right padding, so the plot remains uncluttered.

7. Limitations

The primary limitation of the model is expected-return estimation. Even when covariance inputs are improved through denoising, small changes in expected returns can still produce different portfolio allocations and affect OOS calibration.

The model also relies on historical data to estimate covariance, return distributions, and previous regimes. As a result, it may not fully capture structural breaks, shifting macroeconomic conditions, change in regimes, or stressed market environments in which historical relationships become unstable. For use of 1 year of data, I may have captured regime-specific behavior rather than equilibrium undermining a proper stabilization intent.

Several practical frictions are not yet incorporated into the current backtests. These include transaction costs, slippage, taxes, and explicit liquidity constraints, all of which would likely reduce realized performance and could change the relative attractiveness of more active or concentrated allocations.

The Marchenko-Pastur denoising procedure assumes returns are drawn from an Independent and Identically Distributed (IID) distribution with stable correlations. However, financial correlations are time-varying and/or regime-varying. During stressed market swings correlations could spike causing the signal eigenvalues to shift. In these periods, the Marchenko-Pastur equation may incorrectly classify signals as noise or vice versa. This can create inconsistency and future work can explore regime-conditional denoising or time-varying eigenvalue filtering.

The framework does not incorporate multi-factor expected-return priors, shrinkage-based covariance alternatives, or volatility managed overlays, all of which represent logical next steps for future development and research.

The rolling window backtests further highlight survivorship bias as a constraint because the universe is screened as of January 2024, tickers that failed or were delisted prior to that date are excluded from the formation years which likely can inflate realized returns for earlier periods. Additionally, transaction costs and rebalancing friction are not modeled in the backtests, quarterly rebalancing results represent gross returns only, and slippage on quarterly turnover would compress the rebalancing advantage over buy-and-hold.

These limitations define the scope of the current model. At this stage, it should be viewed as a research framework for improving allocation robustness rather than as a production-ready investment process.

8. Conclusion and Final Thoughts

My project developed a portfolio construction framework that integrates constrained optimization, correlation denoising, implementation-aware allocation, and regime-based simulation into a single research process. The resulting system makes classical mean-variance optimization more practical by incorporating realistic portfolio constraints, cleaner covariance inputs, and distribution-based risk evaluation.

OOS testing showed that the main bottleneck was not the optimization routine itself, but the instability of expected-return estimates. While the model frequently generated positive realized outcomes, baseline return estimators systematically overstated expected performance, reducing forecast calibration and limiting the reliability of the resulting allocations.

In the 4-year rolling window test, the model generated positive median realized returns in 3 of 4 formation years, with quarter rebalancing improving bear-market performance by approximately 14 percentage points in 2022.

Among the estimators tested, Black-Litterman produced the strongest balance between calibration and economic usefulness. This suggests that future improvements are more likely to come from better return priors, stronger robustness testing, and more realistic implementation assumptions than from increasing optimization complexity alone.

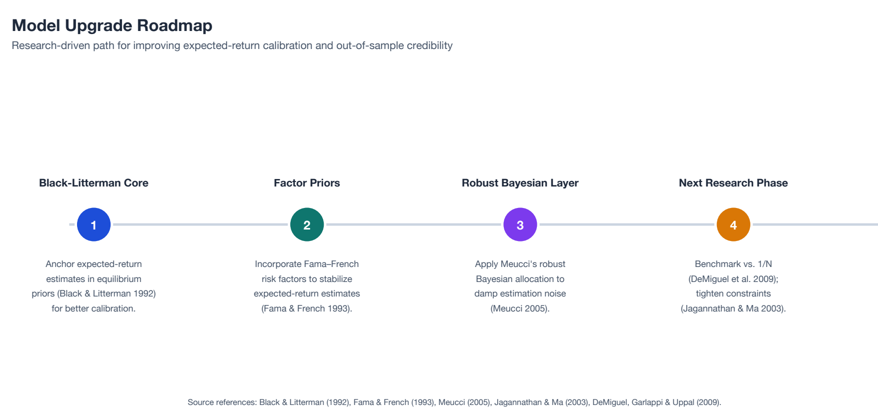


Figure 11

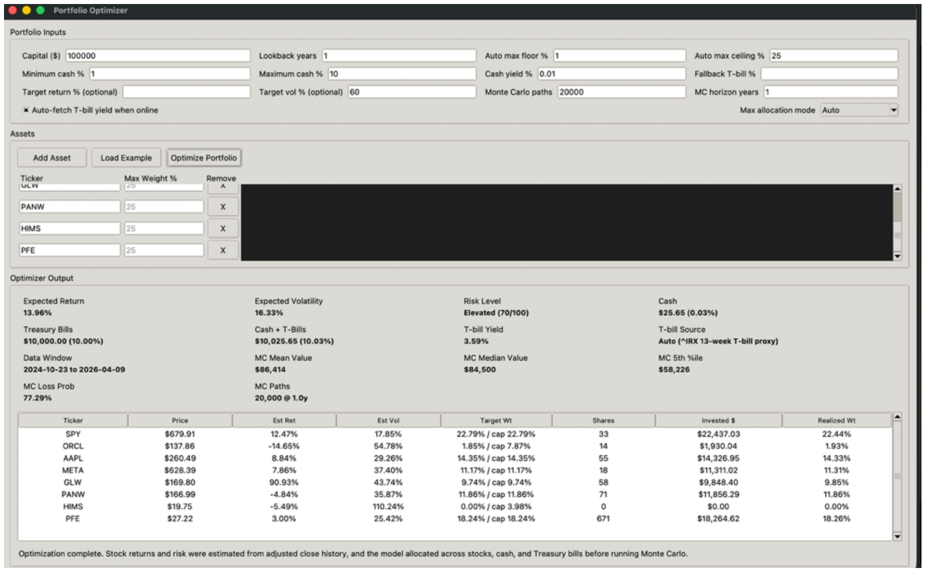
Model development roadmap showing profession from baseline optimization to improved estimation and simulation models.

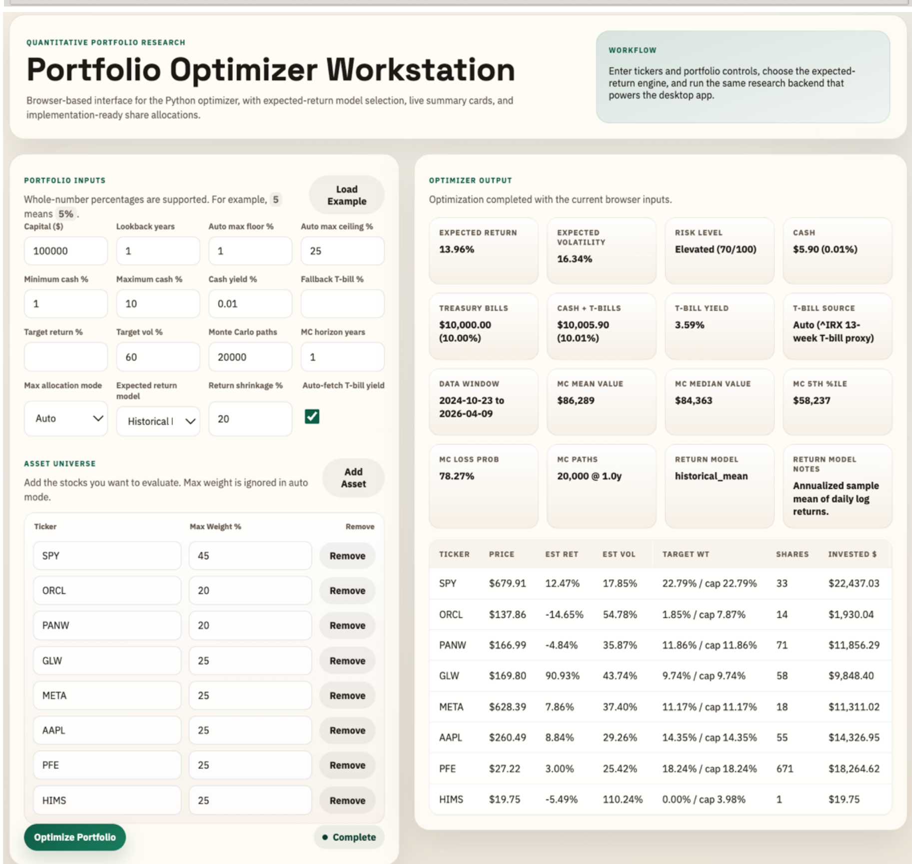
Figure 11 visually summarizes the upgrade roadmap outlined above: Black-Litterman anchors the return layer, followed by Fama-French factor priors to stabilize expectations, Meucci's robust Bayesian allocation to damp estimation noise, and finally benchmarked constraint tuning plus volatility/regime-neutral experimentation tied to Jagannathan & Ma and DeMiguel et al. Each stage references the cited papers so the roadmap can be quoted directly in the methodology section.

Overall, the project supports a practical conclusion: portfolio optimization is most useful when treated as a disciplined portfolio construction framework under uncertainty rather than as a mechanical return-maximization tool. The current model provides a solid foundation for future work in factor-based estimation, covariance shrinkage, transaction-cost-aware optimization, and broader robustness analysis.

Author's Note: I had a lot of fun stepping into the first stage of my research interests in finance. I was able to test myself mentally as I had to learn many new topics especially in mathematics and developing stronger reading skills. Fortunately, I was able to utilize AI tools to assist in understanding technical concepts (especially the mathematics and theories), generating code for the GUI and data visuals for this project, and overall accelerating the research process. I would also like to thank Professor Oommen Thomas and Jordan Schwartzman for their advice throughout the development of this paper.

9. Visuals of GUIs and GitHub





GitHub Link: https://github.com/mateoflo/MateoFlorsheimPortfolioOptimizationModel

10. References

1. Markowitz, Harry. "Portfolio Selection." *The Journal of Finance*, vol. 7, no. 1, 1952, pp. 77–91.
2. Ledoit, Olivier, and Michael Wolf. "Honey, I Shrunk the Sample Covariance Matrix." *The Journal of Portfolio Management*, vol. 30, no. 4, 2004, pp. 110–119.
3. Jagannathan, Ravi, and Tongshu Ma. "Risk Reduction in Large Portfolios: Why Imposing the Wrong Constraints Helps." *The Journal of Finance*, vol. 58, no. 4, 2003, pp. 1651–1684.
4. Jorion, Philippe. "Bayes-Stein Estimation for Portfolio Analysis." *The Journal of Financial and Quantitative Analysis*, vol. 21, no. 3, 1986, pp. 279–292.
5. Black, Fischer, and Robert Litterman. "Global Portfolio Optimization." *Financial Analysts Journal*, vol. 48, no. 5, 1992, pp. 28–43.
6. Fama, Eugene F., and Kenneth R. French. "Common Risk Factors in the Returns on Stocks and Bonds." *Journal of Financial Economics*, vol. 33, no. 1, 1993, pp. 3–56.
7. DeMiguel, Victor, Lorenzo Garlappi, and Raman Uppal. "Optimal Versus Naive Diversification: How Inefficient Is the 1/N Portfolio Strategy?" *The Review of Financial Studies*, vol. 22, no. 5, 2009, pp. 1915–1953.
8. Hamilton, James D. "A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle." *Econometrica*, vol. 57, no. 2, 1989, pp. 357–384.
9. Laloux, Laurent, Pierre Cizeau, Marc Potters, and Jean-Philippe Bouchaud. "Noise Dressing of Financial Correlation Matrices." *Physical Review Letters*, vol. 83, no. 7, 1999, pp. 1467–1470.
10. Marchenko, V. A., and L. A. Pastur. "Distribution of Eigenvalues for Some Sets of Random Matrices." *Mathematics of the USSR-Sbornik*, vol. 1, no. 4, 1967, pp. 457–483.
11. Moreira, Alan, and Tyler Muir. "Volatility Managed Portfolios." *The Journal of Finance*, vol. 72, no. 4, 2017, pp. 1611–1644.
12. Meucci, Attilio. *Risk and Asset Allocation*. Springer, 2005.

Disclaimer: AI tools were used to assist me in the development of the writing of code, specifically the GUIs and Web interfaces, as well as the development of the SVG visuals.