STAT 4830
Final Presentation

Portfolio Refinement Through Iterative Sequential Modeling (PRISM)

Team PRISM:
Dhruv Gupta, Aiden Lee, Frank Ma,
Kelly Wang, Didrik Wiig-Andersen

Wharton
University of Pennsylvania

The Stock Market Problem

Uncertainty in the Market

[Picture from WSJ]

Motivation

Can we develop an algorithm that is robust to market shocks?

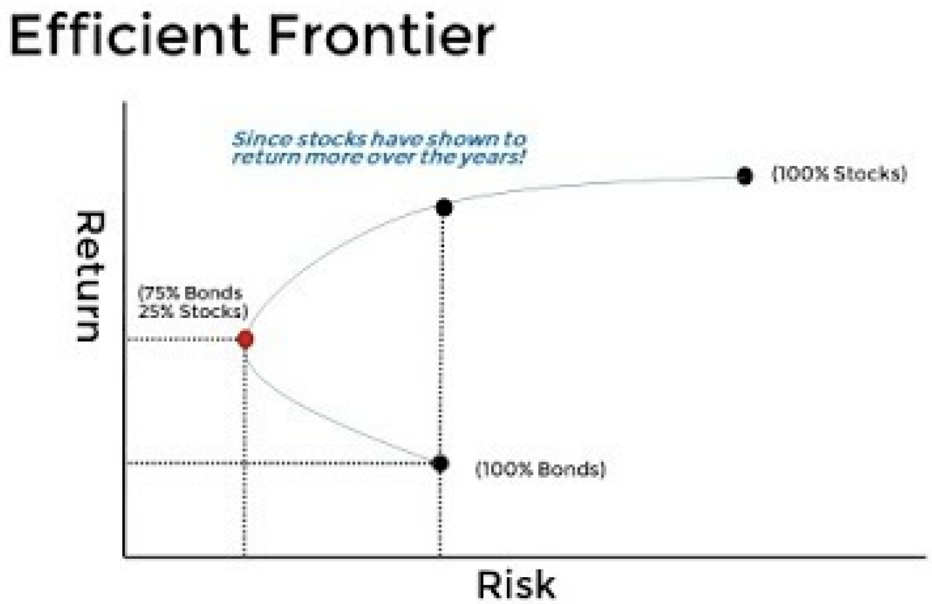
Our Goal: Optimize a daily portfolio of assets to maximize risk-adjusted returns while incorporating penalties that limit:

- drawdown (protecting against significant losses)
- turnover (maintaining portfolio stability)
- concentration risk (ensuring proper diversification)

Measurable Outcome:
Generating returns superior to a given baseline ("safe") portfolio.

Mean-Variance Optimization (Markowitz, 1952)

- Balance expected return against portfolio variance.
- Core insight - combining assets with imperfect correlations reduces risk and can rise expected return.
- Led to the discovery of the efficient frontier.



Sharpe Ratio (Sharpe, 1966)

$$SR = \left(\frac{R_t - r_{f,t}}{\sigma_t} \right)$$

- Shows that maximizing the Sharpe ratio selects the "tangency portfolio" – the point on the efficient frontier with the highest risk-adjusted return.
- Employed as one of many inputs in real-world portfolio construction.

Conditional Value-at-Risk (CVaR): "Expected Shortfall"

- Measures the average loss in the worst $\alpha\%$ of outcomes (i.e. "in the worst 5% of days, how much do I lose on average?")
- **Rockafellar and Uryasev (2000):** developed an approach to maximize Sharpe subject to $CVaR \leq X$.
- Widely adopted after the 2008 crisis to build more shock-resilient portfolios.

Technical Approach

Modeling

Database	Query
Web of Science	("stock" OR "equity") AND ("portfolio optimization" OR "asset allocation") AND ("risk measures" OR "risk metrics") AND ("drawdown" OR "value-at-risk" OR "conditional value-at-risk" OR liquidity OR skewness OR kurtosis) AND (optimization OR "gradient descent" OR "adam" OR "rmsprop")
Scopus	TITLE-ABS-KEY("portfolio optimization" OR "asset allocation") AND ("risk measures" OR "risk metrics") AND ("drawdown" OR "value-at-risk" OR "conditional value-at-risk" OR liquidity OR skewness OR kurtosis) AND ("stock" OR "equity")
EconLit	("portfolio optimization" OR "portfolio selection") AND ("risk measures" OR "risk metrics" OR "alternative risk premia") AND ("drawdown" OR "value-at-risk" OR "conditional value-at-risk" OR "tail risk" OR "drawdown range" OR "short selling") AND ("liquidity" OR "factor-based" OR "multi-factor" OR "factor selection" OR "factor exposure")

- 177 articles across Web of Science, Scopus and EconLit.
- **Four Key Metrics:** Sortino Ratio, Maximum Drawdown, Turnover, and Concentration Penalty.

Formulation

Objective Function:

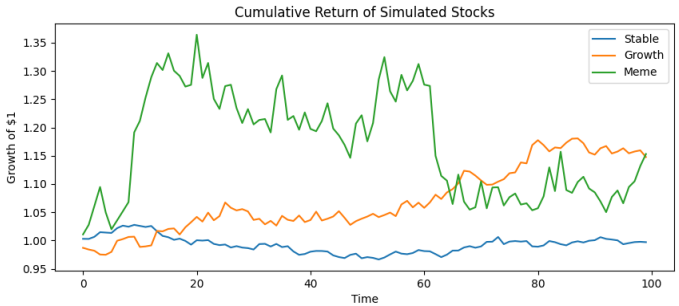
$$\max_{\mathbf{w}_t} \quad \alpha_1 \cdot \text{Sortino}_t(\mathbf{R}_p, \mathbf{r}_f) - \alpha_2 \cdot \text{MaxDD}_t(\mathbf{R}_p)$$
$$- \alpha_3 \cdot \text{Turnover}(\mathbf{w}_t, \mathbf{w}_{t-1}) - \alpha_4 \cdot \text{CP}(\mathbf{w}_t)$$

Constraints:

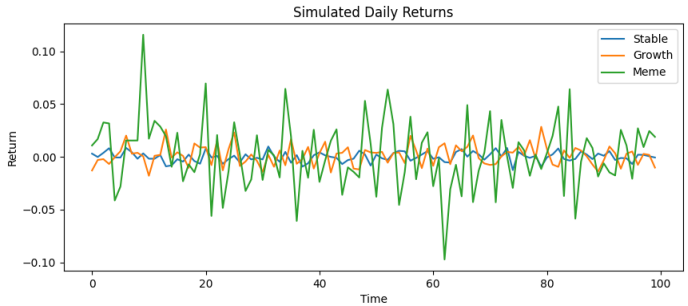
$$\sum_{i=1}^{N} w_{t,i} = 1 \quad \text{where } w_{t,i} \geq 0 \, \forall t, i$$

Visualization

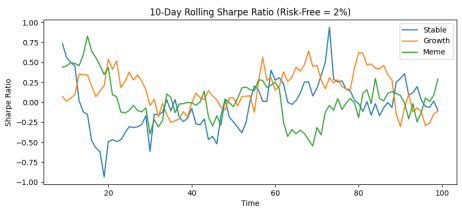
Risk and Return Metrics



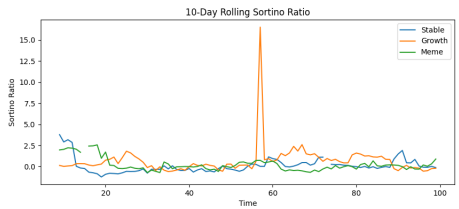
Cumulative Returns



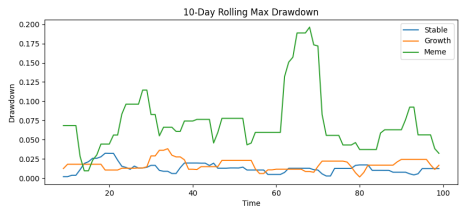
Daily Returns



Sharpe Ratio



Sortino Ratio



Max Drawdown

Variable Definitions

Weighted Portfolio's Returns:

$$R_p = \sum_{i=1}^{n} w_{i,t} \cdot R_{i,t}$$

$n \in \mathbb{N}$	Number of assets held in the portfolio
$w_{i,t} \in \mathbb{R}$	Weight for asset i at time t
$R_{i,t} \in \mathbb{R}$	Asset i's return at time t
$r_{f,t} \in \mathbb{R}$	Risk-free rate at time t
$\alpha_j \in \mathbb{R}_+$	Objective weights, $j \in \{1,2,3,4\}$
$m \in \mathbb{R}_+$	Window size for historical calculation
$\varepsilon \in \mathbb{R}_+$	Small constant for numerical stability

Sortino Ratio

Obj. Func. (1st Term)

$$\text{Sortino}_t(\mathbf{R}_p, \mathbf{r}_f) = \frac{\mathbb{E}[\mathbf{R}_p - \mathbf{r}_f]}{\sigma_{\text{downside}} + \varepsilon} \tag{1}$$

$$\text{where} \quad \sigma_{\text{downside}} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \min(R_p - r_{f,t}, \text{NA})^2} \tag{2}$$

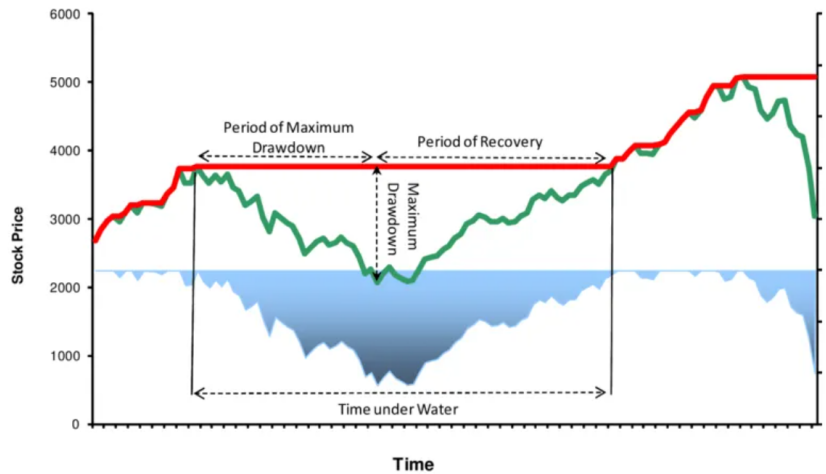
(Calculated based on data from the past m days, i.e. $\mathbf{R}_p$, $\mathbf{r}_f \in \mathbb{R}^m$.)

Maximum Drawdown

Obj. Func. (2nd Term)

$$\text{MaxDD}_t(\mathbf{R}_p) = \frac{\max_{T\in[t-m,t]} \text{CR}_T - \text{CR}_t}{\max_{T\in[t-m,t]} \text{CR}_T + \varepsilon} \tag{3}$$

$$\text{where} \quad \text{CR}_t = \prod_{i=t-m}^{t} (1 + R_i) \tag{4}$$



(Calculated based on data from the past m days, i.e. $\mathbf{R}_p \in \mathbb{R}^m$.)

Turnover Term

Obj. Func. (3rd Term)

Proxy to represent transaction costs (for $\mathbf{w}_t \in \mathbb{R}^n$):

$$\text{Turnover}(\mathbf{w}_t, \mathbf{w}_{t-1}) = \frac{1}{2} \sum_{i=1}^{N} |w_{i,t} - w_{i,t-1}| \tag{5}$$

Concentration Penalty

Obj. Func. (4th Term)

$$\begin{aligned} \mathrm{CP}(\mathbf{w}_t) = &\max(\mathrm{ENP}_{\min} - \mathrm{ENP}(\mathbf{w}_t), 0) \\ &+ \max(\mathrm{ENP}(\mathbf{w}_t) - \mathrm{ENP}_{\max}, 0) \end{aligned} \tag{6}$$

Effective Number of Positions (for $\mathbf{w}_t \in \mathbb{R}^n$):

$$\mathrm{ENP}(\mathbf{w}_t) = \frac{1}{\mathrm{HHI}(\mathbf{w}_t) + \varepsilon} \tag{7}$$

$$\mathrm{HHI}(\mathbf{w}_t) = \sum_{i=1}^{N} (w_{i,t})^2 \tag{8}$$

Methodology

Algorithm Selection

- **Online Gradient Descent (OGD)**
  - Sequential optimization adapting to changing market conditions
  - No need to retrain on historical data - continuous learning
  - Differentiable objective function with automatic gradient computation
- **Connection to Course Concepts**
  - Gradient-based optimization with constraints
  - Multi-objective function balancing competing financial goals
  - Online learning for sequential decision making

Methodology

Tuning Procedure

- **Critical Hyperparameters**
  - Objective weights (α_i) - balance risk/return tradeoff
  - Window size - determines optimization lookback period
  - Learning rate - controls adaptation speed
  - ENP constraints - enforce diversification bounds
- **Hyperparameter Exploration**
  - Window sizes: 5, 21, 63, 252 days (1wk, 1mo, 3mo, 1yr)
  - Objective weights: grid search prioritizing Sortino and drawdown control
  - ENP constraints: calibrated to allow sector concentration while preventing single-stock dominance
  - Learning rates: tested 0.1-1.0 to balance stability and responsiveness

Implementation

Key Choices

- **PyTorch Framework**
  - Automatic differentiation for gradient computation
  - Built-in optimizers with configurable learning rates
- **Implementation Details**
  - Softmax normalization to enforce $\sum_i w_i = 1$ constraint
  - Rolling window implementation for adaptive optimization
  - Tried out an SQL query system

Implementation

Challenges & Adaptations

- **Computational Challenges**
  - Scalability with large asset universes (> 500 stocks)
  - Memory constraints when tracking long optimization histories
- **Adaptations & Solutions**
  - Adjustable objective weights to control balance between goals
  - Portfolio concentration constraints to manage diversification
  - Epsilon factors to prevent division by zero in metric calculations

Demonstration

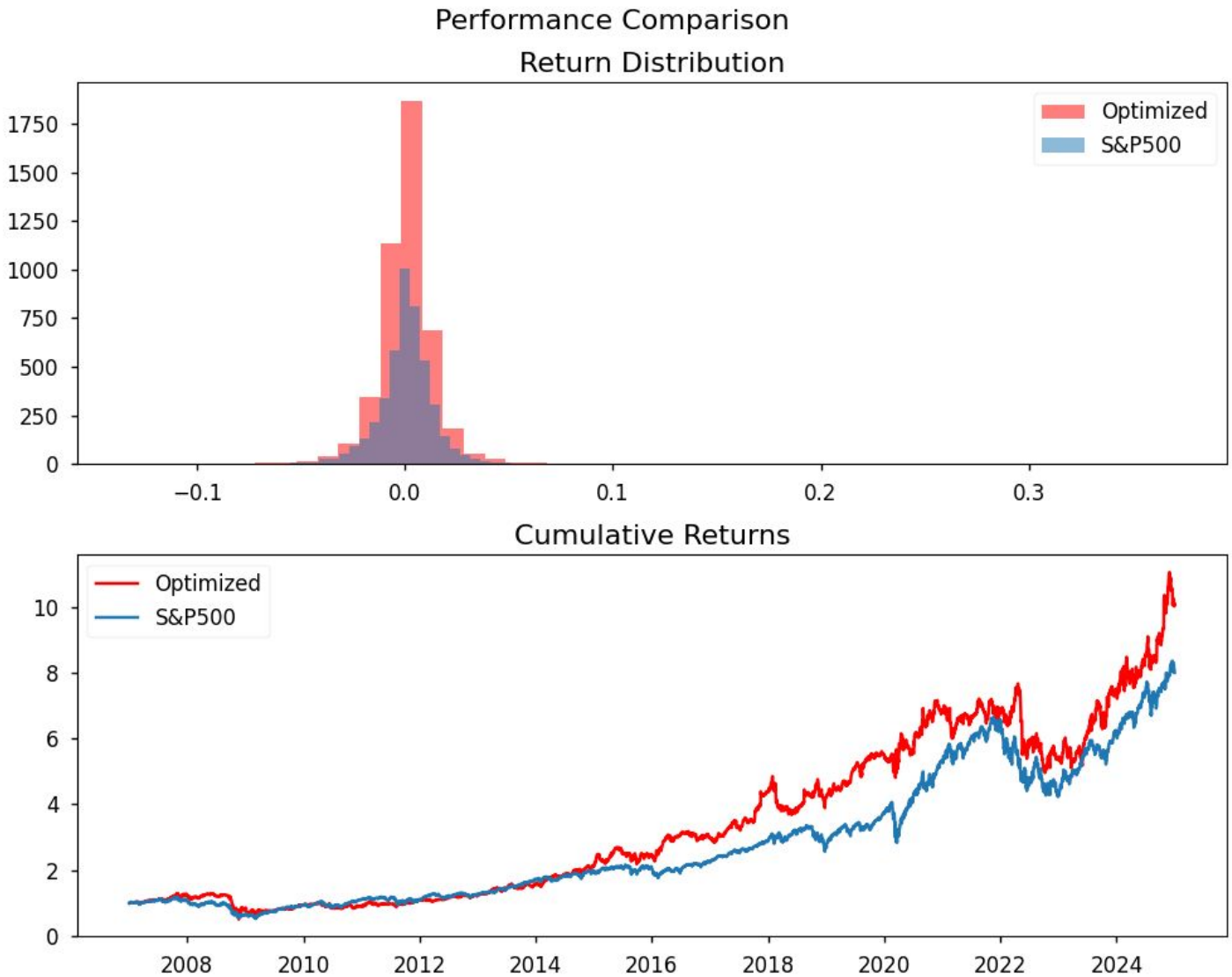
Visuals

Presenting our key quantitative results:

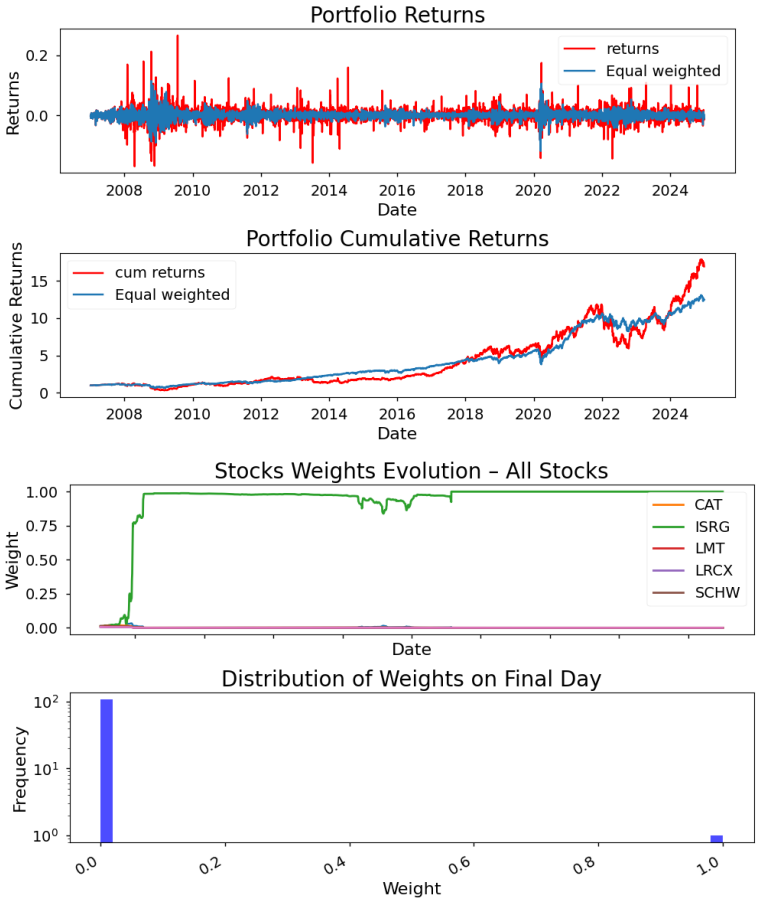
https://droov-opt.hf.space/

Final Output

Graphs



Intermediate Results

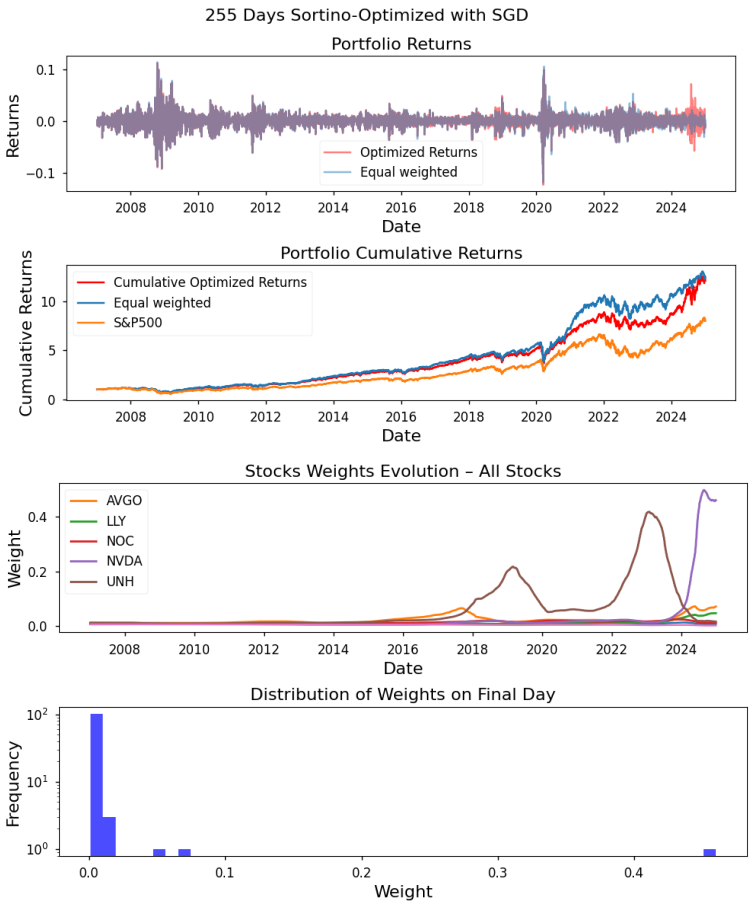


Observations

- Portfolio outperforms equal-weighted benchmark, especially post-2020.
- Optimization leads to highly concentrated positions (e.g., single stock weight near 100%).
- Weight distribution on final day reveals near-binary allocation: almost all in one stock.
- We were actually more volatile than market –¿ did we really optimize sharpe?

Intermediate Results - Optimizing Sortino Only

Graphs

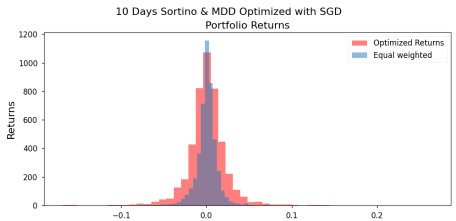


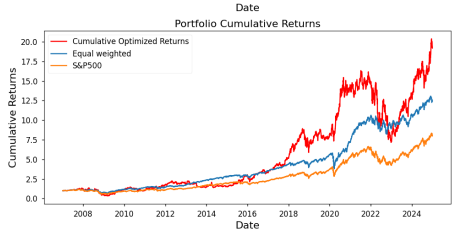
Observations

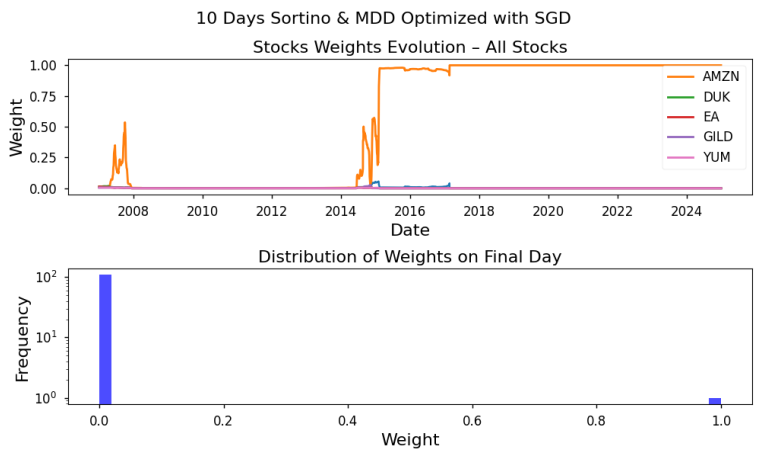
- We are overperforming equal-weighting, but outperforming S&P.
- Optimization leads to exposure across several stocks, but still relatively concentrated (e.g., NOC, UNH).
- Unclear whether we are more or less volatile than market.

Intermediate Results - Optimizing Sortino and MDD

Graphs





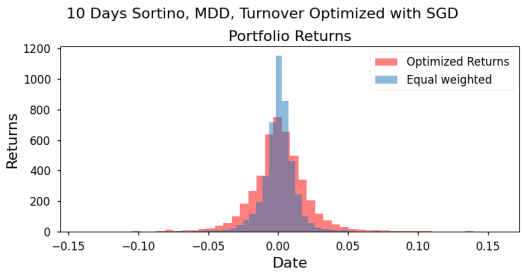


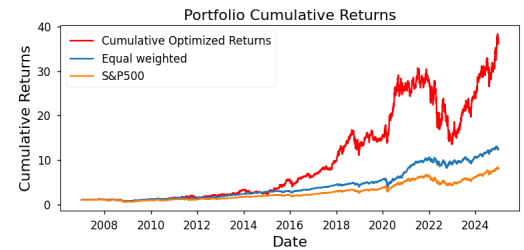
Observations

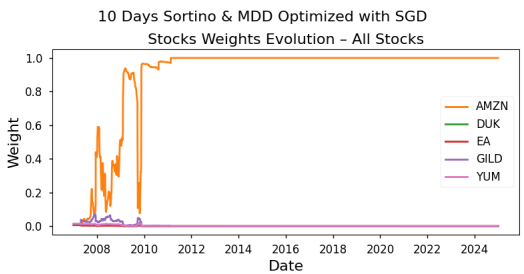
- 10 days was way too volatile, even with max drawdown penalty
- Portfolio cumulative returns outperform S&P500 and equal-weighted benchmarks, but at what cost...
- Allocation is again highly concentrated — majority in AMZN for extended periods.
- Weight distribution still shows heavy tail: one dominant stock, many with near-zero allocation.

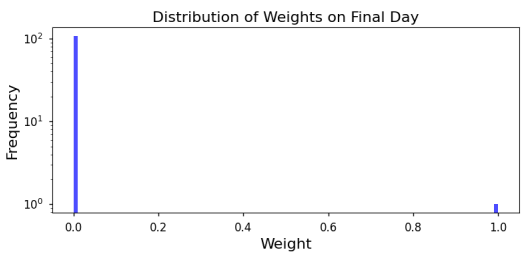
Intermediate Results - Optimizing Sortino, MDD, Turnover

Graphs







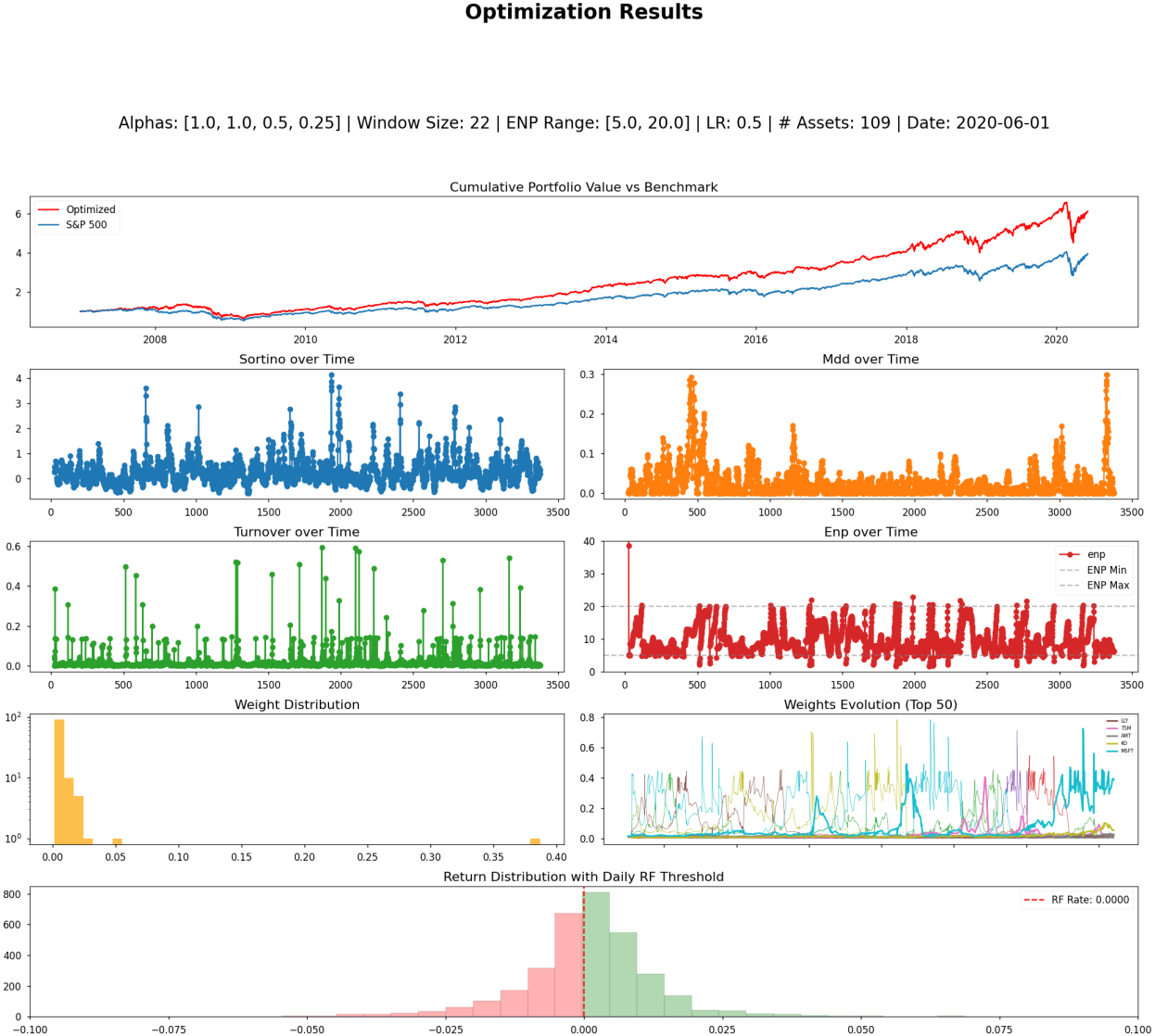


Observations

- Portfolio maintains strong outperformance vs. benchmarks.
- Turnover constraint reduces trading frequency but maintains exposure to dominant stocks.
- Concentration issue persists

Intermediate Results

Full Objective Function



Intermediate Results - Hyperparameters

Graphs

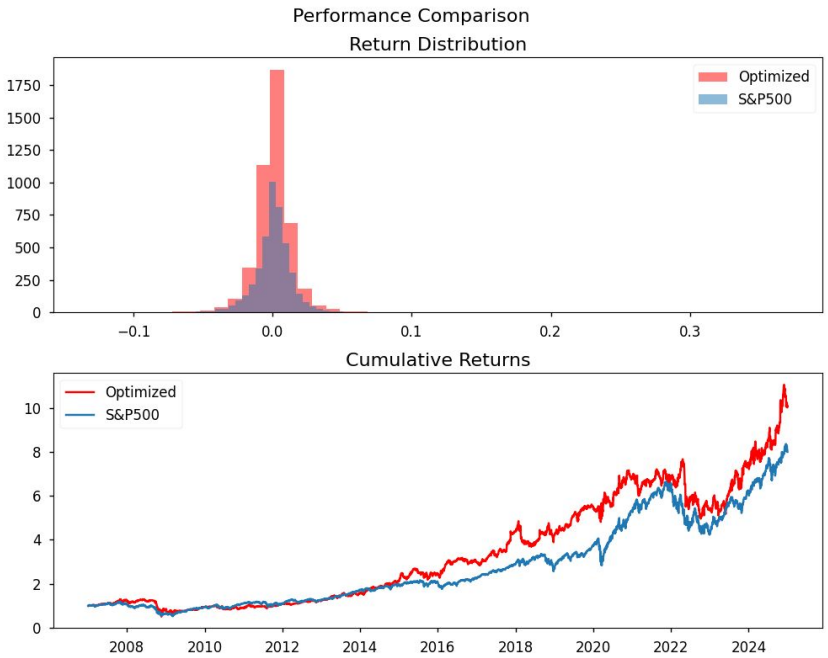
```
 Optimal Hyperparameters:
Window Size     : 181
Learning Rate   : 0.8655
Alpha Weights   : [1.0, 6.080, 0.935, 5.154]
Best Sortino    : 0.0958
```

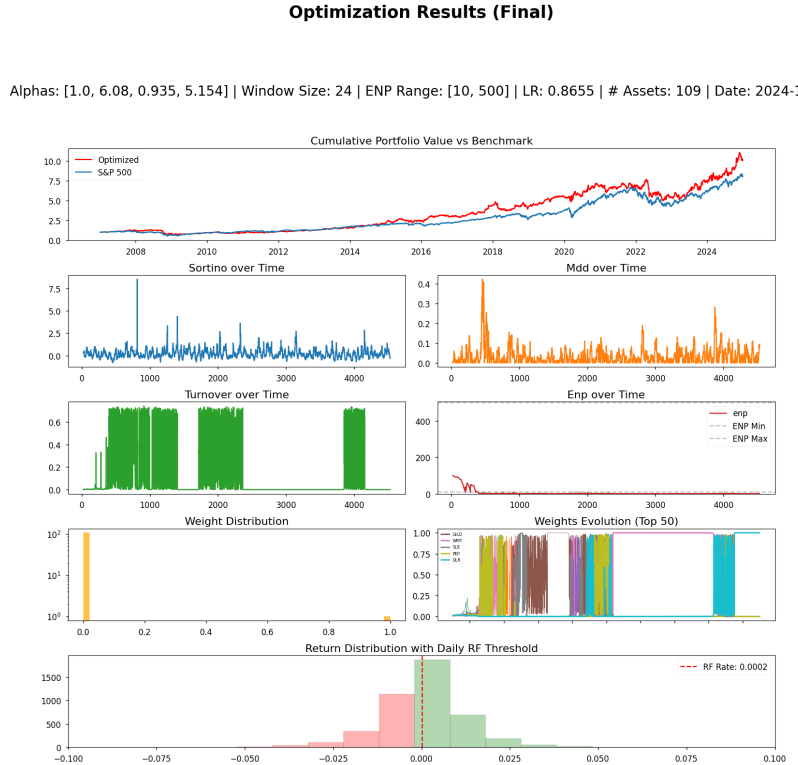
Observations

- 5 tunable hyperparamters
- used 2007-2010 for tuning, and subsequent years for evaluation
- used Gaussian Process Minimize from scikit-optimization
- supposedly optimal hyperparameters looked a little suspicious

Final Results

Graphs





Results

Interpretation & Evaluation

- **Comparing our results to baseline methods:**
  - Improvements made moving from Sharpe $\rightarrow$ Sortino $\rightarrow$ our final objective function.
  - We also compare ourselves to the S&P500 and an equally weighted portfolio of our universe of stocks.
  - We iteratively made improvements to baseline based on qualitative metrics, but we did not actually have a systematic way to compare different optimizations
- **Interpretation:**
  - The supposedly optimized strategy had an all-history Sharpe of 0.0363, max drawdown of 0.606, and annualized excess return of 7.44% annually
  - Although the results didn't look mindblowing, we are earning a decent return not attributable to market!

Reflections

- **Primary Challenge**
  - Deciding and optimizing objective weights
  - Deciding on benchmarks and criteria for evaluating our strategy
- **Expectations**
  - Getting access to financial data was surprisingly easy to integrate into our code.
  - Choosing our universe of stocks was difficult for implementation and justification reasons.
  - Migration to PyTorch took more time than expected.
- **Evolution**
  - Lit review - simple 7-stock model with sortino, MDD using numpy - stock universe expansion - pytorch migration, additional objectives like turnover and ENP
- **AI Assistance**
  - Claude helped write the first versions of our implementations.
  - ChatGPT helped refine our project outline and trajectory for improvements

Individual Contributions

Dhruv (Slide 1)

- I think generally just how well OGD worked? I was surprised because my initial idea was training a model on some window size of returns data and use that to predict for stocks (regardless of previous data) so it was cool to see that a simpler idea like OGD was still effective
- I think I just enjoyed the lectures where we went through gradient descent and the different hyperparams in a lot of detail. That really helped with my conceptual understanding
- I came into this not even knowing what gradient descent really was. I think seeing it in progress has been super useful, and I even used it in Chicago Trading Competition a couple weeks ago to come 5th out of the 41 teams.

Individual Contributions

Dhruv (Slide 2)

- Ever since I started working on the demo I've really enjoyed that part. I'd look to try and get even more tickers, adjust it to the point where we are able to get live data, and see if, given some set of tickers, we could implement grid search for hyperparams within the demo too.
- I think that I would have liked to get onto using Torch from the start, and actually implementing auto grad instead of doing manual gradient calculation. If we had that sort of baseline to build off of from the start, I think we would be 2-3 weeks ahead of where we were, because when we made that switch we lost a lot of time

Individual Contributions

Aiden (Slide 1)

- The incorporation of multiple objectives actually does work and performs better than S&P.
- Lectures on how gradient descent works, both the implementation and theory behind it, helped understanding our online gradient descent at a deeper level.
- I initially thought optimization was a very strict and theoretical field (i.e. convex optimization). However, I learned that it's very flexible and can accommodate many different types of objective functions, and running gradient descent works surprisingly well on many different tasks, even if it's not optimal.

Individual Contributions

Aiden (Slide 2)

- I would try incorporating better compute resources to test on a wider range of stocks, since our current stock universe is still very limited. Also we could potentially expand into derivatives or other types of tradable assets.
- We would have started implementing directly with PyTorch and have a more consistent evaluation method for our algorithm throughout the process.

Individual Contributions

Frank (Slide 1)

- What was the most surprising result or finding during the project?
  - I think the whole idea about online gradient descent was surprising to me. I have not heard about OGD until this project, and I was surprised online gradient descent could work
  - I was also surprised by the results from the hyperparameter tuning. It seemed very counterintuitive that the supposedly optimal hyperparameters resulted in wild fluctuations in weights because the learning rate was set super high

Individual Contributions

Frank (Slide 2)

- Which specific lecture topic, concept, or technique from the course was most useful for your project? Explain how.
  - The lecture about computational graphs and common pitfalls in auto diff was the most helpful. When trying to get OGD to work with auto diff, I initially ran into a bunch of bugs because of an inplace operation I accidentally wrote in.
  - (It was dumb because the inplace operation wasnt even for gradient descent, it was related to the live visualization during optimization that I wanted to include)

Individual Contributions

Frank (Slide 3)

- How has your perspective on applying optimization in practice changed since the beginning of the course?
  - To be very honest I did not know what optimization meant before coming into the course. The email example from class 0 was the very first time I wrote an optimization.
  - Now I understand how impressive torch is. I also feel a little more confident working with torch and optimization. I am currently reworking a previous internship project on revenue forecasting with MLE and rewriting it in torch so that I get more control over the model.
  - In general I think I am comfortable writing some optimization for an internship / for a project. Or at least I know where I can get started and what changes I can make to a baseline optimization

Individual Contributions

Frank (Slide 4)

- If you had two more weeks, what specific next step would you take on the project?
  - I would actually establish a common criteria for strategy evaluation (CAPM alpha), and evaluate each of the interactions against that benchmark
  - I also want to maybe use another data source, get data for 2025 up to April, make the data use the same schema as WRDS data, and let our model run on 2025 data
  - I would expand the universe of stocks other than all US stocks
- If you could restart the project, what is the single biggest change you would make to your approach or plan?
  - Instead of learning the weights, I think it might be interesting to learn the mapping of stock features to an optimal weight (and then project the weights back onto simplex)

Individual Contributions

Kelly (Slide 1)

- I was most surprised to see how effective the evolution of our objective function was in impacting our results. I knew the baseline function of using the Sharpe ratio, which is the only metric taught in introductory finance courses to improve one's portfolio, wasn't the most efficient method. Moving along to bringing in the Sortino ratio and Maximum Drawdown function along with transaction costs made noticeable improvements to our returns. Given that our algorithm was relatively simplistic, adding these theoretically intuitive additions to our project was surprisingly easy to integrate into our code. I was expecting a more difficult time in developing an improved formulation of the base model.
- Because our project relies on a version of gradient descent, I felt the lecture on "How to compute gradients in PyTorch" was the most useful topic for our implementation. Running through an example that used PyTorch as its code implementation was the basis for how we structured our baseline algorithm. The lecture provided intuition behind how the code should work, which allowed us to debug anything we didn't think was working.

Individual Contributions

Kelly (Slide 2)

- Before this course I was under the assumption that pure optimization algorithms required the problem to be structured nicely such that all preliminary assumptions are met for algorithms to run. Even though these assumptions still persist, I've learned that fulfilling these assumptions can be a lot more flexible, because the results are still decent even if you can't verify all the properties of a complex optimization problem.
- Given two more weeks, I would try to incorporate short selling into our code (i.e., allowing weights to become negative). Though short selling introduces more risk in the real world, I would be interested to see how we could improve our code such that short-selling could actually be used to hedge certain risk that we plan to take on such that our overall portfolio's returns are less susceptible to shocks in the market.
- If I restarted this project, I would now take more time to the characteristics of our problem to better understand how we can systematically create a multi-objective function. More specifically, by having a better understanding of how the change in weights impact the variability and return of our portfolio, we could have had more time to explore other online implementation techniques like mirror descent or multiplicative weights.

Individual Contributions

Didrik (Slide 1)

- The most surprising finding was how concentration risk dominated our portfolio optimization despite our attempts to control it. Even with explicit penalties in our objective function, the algorithm consistently allocated heavily to tech stocks like AMZN and NVDA. I think this showed how the tension between return maximization and risk management is more challenging than theoretical models suggest, and reinforced why real-world portfolios need explicit diversification constraints.

Individual Contributions

Didrik (Slide 2)

- I think the lecture on "How to compute gradients in PyTorch" was important for our implementation. The learnings from the class allowed us to experiment with more complex objective functions without getting slowed down in overly complex calculations, especially for non-trivial components like maximum drawdown which involves non-smooth operations.

Individual Contributions

Didrik (Slide 3)

- I have gained a much deeper appreciation for the gap between theoretical optimization and practical implementation. Before the course, I viewed optimization as primarily a mathematical exercise with clean, deterministic solutions. Now I understand it as a more iterative, experimental process that requires balancing competing objectives, handling noisy data, and making trade-offs. The stochastic nature of financial markets makes this particularly evident, where optimal solutions on historical data can perform poorly on new data.

Individual Contributions (Slide 4)

Didrik (Slide 4)

- I would implement a robust backtesting framework that evaluates our strategy across different market regimes (bull markets, bear markets, high/low volatility periods, large shocks like Covid-19). Currently, our evaluation is somewhat ad-hoc, making it difficult to systematically compare different parameter settings. A formal backtesting approach would help us understand when our strategy underperforms and potentially incorporate components that adapt hyperparameters based on market conditions.

Individual Contributions

Didrik (Slide 5)

- Similar to what Dhruv and Aiden mentioned, I would start with PyTorch from day one rather than implementing gradients manually. The time we spent debugging gradient calculations could have been better used exploring different objective formulations and hyperparameter configurations. Additionally, I would establish clear quantitative metrics for success earlier in the project, which would have helped guide our development process more systematically instead of relying on qualitative assessments of performance.