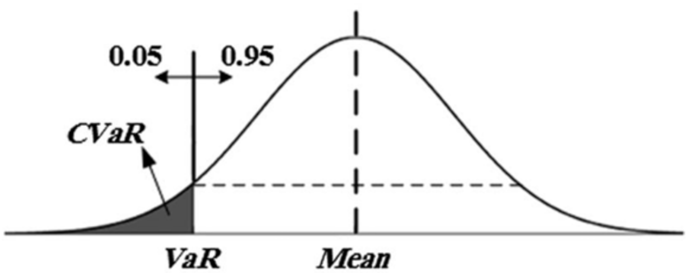
OPTIMIZATION- I

PROJECT 1 - LINEAR PROGRAMMING

Portfolio Optimization using Conditional Value-at-Risk (CVaR)



Prepared by: Abhiroop Kumar, Manny Escalante, Simoni Dalal, Stiles Clements

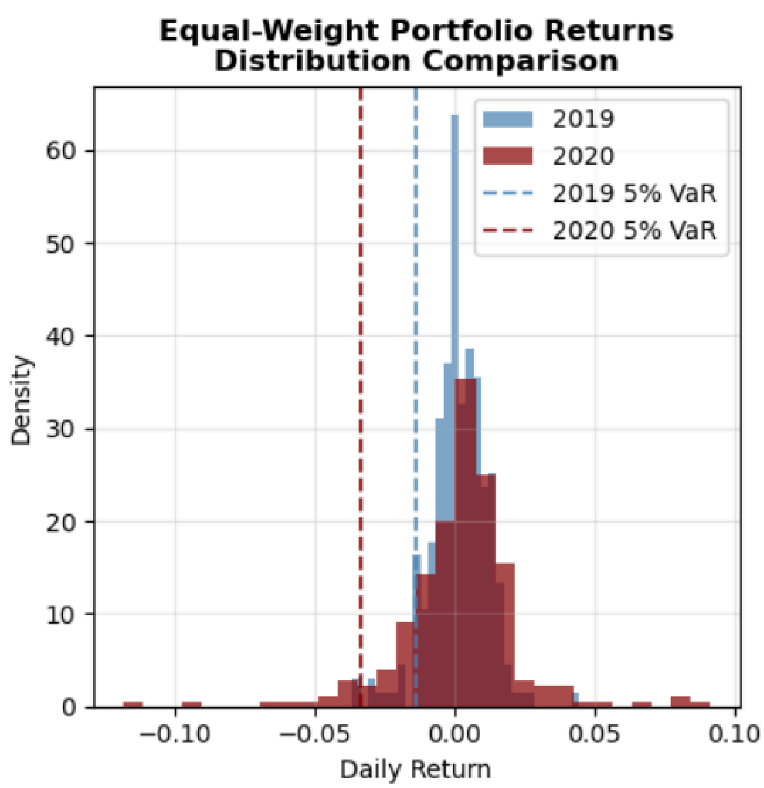
Executive Summary:

- Developed CVaR optimization models for portfolio risk management using 2019-2020 stock data
- Static portfolio achieved CVaR of 0.047 vs. monthly rebalancing average of 0.032 (-31% improvement)
- Monthly rebalancing showed higher volatility (max CVaR: 0.101 in March 2020)
- Portfolio stability analysis revealed weight changes exceeding 5 percentage points threshold. Proposed stability constraints for implementation

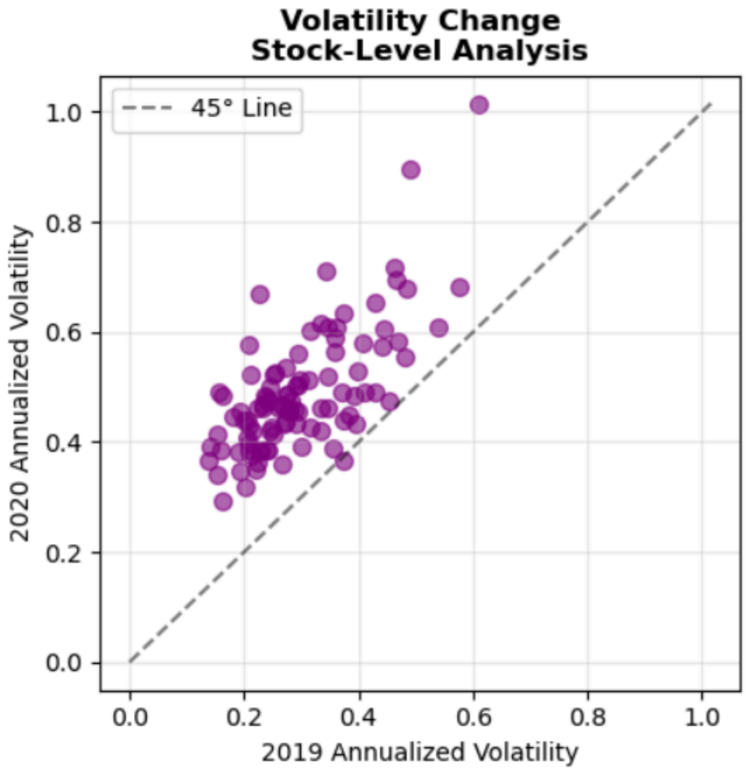
Part 1: Data Analysis and CVaR model implementation

The analysis utilizes daily stock price data from the NASDAQ 100 universe. The first data file is from 2019 (baseline condition) and the second data file is from 2020 (COVID-19 period). This data structure allows us to train our CVaR optimization model on normal Market conditions and validate performance during an extreme market stress event.

All returns are calculated as daily percentage change.



This shows how stock returns changed from 2019 to 2020. In 2019 (blue), most daily returns were small and clustered around zero - calm markets. In 2020 (red), the returns spread out much more with bigger losses and gains - much more volatile markets. The dotted lines show the "5% worst case" - in 2019, the worst 5% of days had losses around 1.8%, but in 2020, the worst days had losses around 4.8%. Basically, 2020 was much riskier.



Each dot represents one stock. The x-axis shows how volatile each stock was in 2019, and the y-axis shows how volatile it was in 2020. If markets stayed the same, all dots would sit on the diagonal line. But most dots are above the line, meaning almost every single stock became more volatile in 2020 than it was in 2019.

2020 was way more dangerous for investors than 2019, which is why the report focuses on building portfolios that can handle these kinds of crisis periods.

Before solving for Part 2, we need to set up the model. The Conditional Value-at-Rick (CVaR) portfolio optimization problem minimizes the expected loss in the worst-case tail scenarios while ensuring expected returns.

```
model = gp.Model("CVaR_Minimization")
```

Decision Variables: The optimization model employs three sets of decision variables:

1. The main decision variables are the portfolio weights, represented as $x = (x_1, x_2, \ldots\ldots x_n)$ where x_j is the weight of each stock.
2. α: Value-at-Risk threshold (95th percentile loss level)

3. Auxiliary variables u_k to linearize the problem and representing losses exceeding VaR in period k

```
x = model.addMVar(n, lb=0, name="x")
alpha = model.addMVar(1, name="alpha")
u = model.addMVar(q, lb=0, name="u")
```

Objective Function: The objective is to minimize the CvaR risk. The linearized objective function is given as

$$\tilde{F}_\beta(\boldsymbol{x}) = \min_{\alpha,u}\ \alpha + \frac{1}{(1-\beta)q}\sum_{k=1}^{q} u_k$$

```
model.setObjective(alpha + (1 / ((1 - beta) * q)) * gp.quicksum(u), sense=gp.GRB.MINIMIZE)
```

Here, α represents the VaR threshold, β is the confidence interval (0.95 for 95% confidence), and q is the number of historical scenarios (the number of rows in the 2019 data).

Constraints: These are the constraints to define the feasible regions

1. Budget constraints: The portfolio weights must sum to 1 i.e a fully invested portfolio

```
model.addConstr(gp.quicksum(x) == 1)
```

2. Return Constraints: The portfolio's expected return must be at least a given amount R. This ensures the portfolio meets minimum return requirements (R= 0.02% daily)

```
model.addConstr(gp.quicksum(mean_returns[j] * x[j] for j in range(n)) >= min_return_R)
```

3. Auxiliary constraints: The auxiliary variables u_k are introduced to linearize the problem and must satisfy the following condition for each historical scenario k

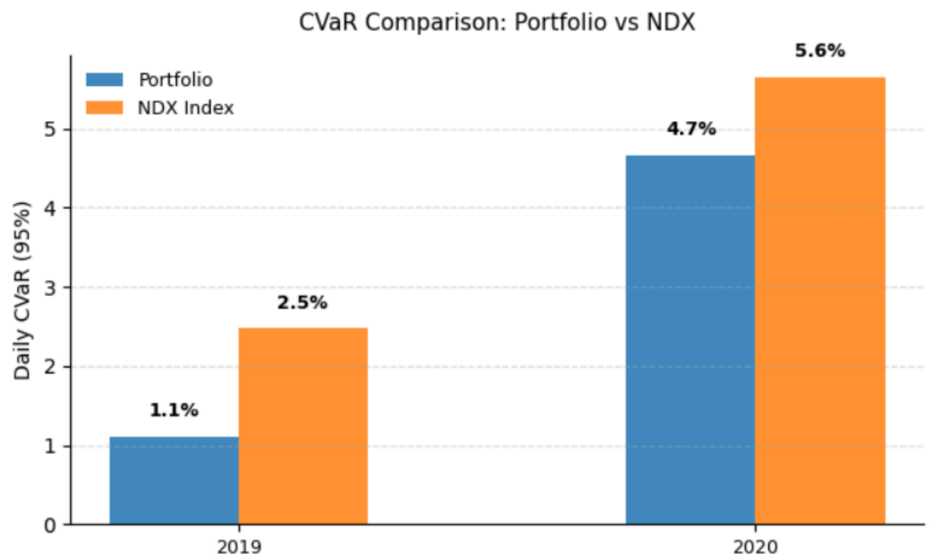
$$u_k \geq -\boldsymbol{x}^T \boldsymbol{y}_k - \alpha$$

```
model.addConstrs(u[k] >= -gp.quicksum(returns_data.iloc[k, j] * x[j] for j in range(n)) - alpha for k in range(q))
```

4. Non-negativity constraint for portfolio weights and auxiliary variables

```
model.optimize()
```

Part 2: Baseline Portfolio Optimization (β = 0.95, R = 0.02%)



Optimized 2019 Portfolio

After running the optimization for our portfolio, the optimal portfolio VaR for 2019 data was 0.008500. This means that 95% of the time, the portfolio will have a 0.85% loss or better. The CVaR for 2019 data was 0.011090 which means the worst 5% of days average about a 1.11% loss.

	weight
XEL	30.3917%
CHTR	26.4965%
AMGN	8.0766%
CTXS	6.5614%
CHKP	6.2696%
EXC	5.2155%
KHC	4.0510%
CSX	3.4312%
COST	3.0352%
EBAY	2.3486%

Out-of-Sample 2020 Performance

The daily average CVaR using the 2019 portfolio in 2020 was found to be 0.046563.

Why was the portfolio's 2020 performance much worse?

The 2019 data represents a 'relatively normal' market period, while 2020 was a volatile year due to the COVID-19 pandemic. The optimal portfolio for 2019 might not be optimal for 2020 due to this non-stationarity. The out-of-sample CVaR in 2020 is expected to be higher as the 2019-optimized portfolio was not designed to handle the extreme adverse conditions of 2020.

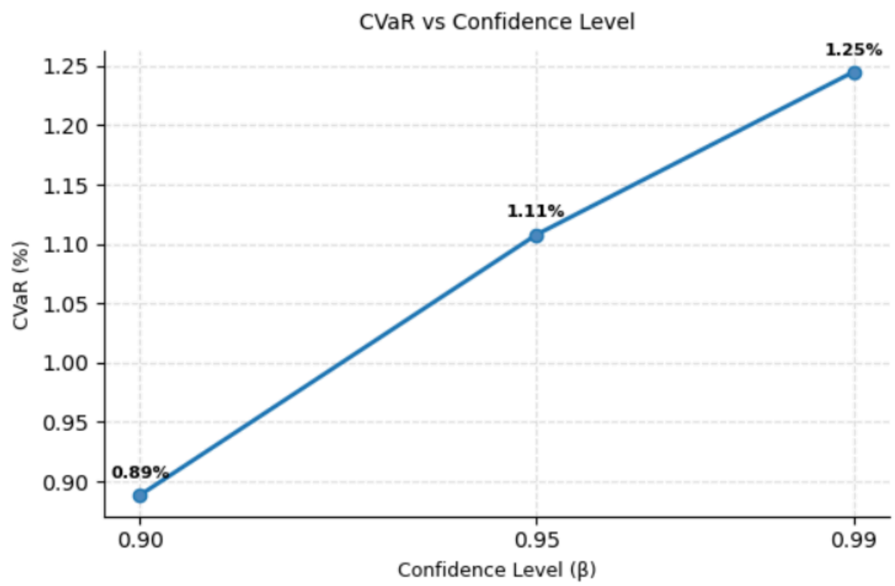
NDX Index Benchmark

To benchmark the optimized portfolio against the broader market, we compare its CVaR with that of the Nasdaq-100 index (NDX). In 2019, the NDX exhibited a CVaR of 2.47%, more than double the optimized portfolio's 1.11%. During the turbulent 2020 period, the NDX's CVaR increased sharply to 5.65%, again substantially higher than the optimized portfolio. This comparison highlights that the optimization procedure meaningfully reduced downside tail risk relative to a passive index strategy, both under normal market conditions and during extreme stress.

Is it a good idea to keep the same portfolio across years?

Whether it is wise to keep the same portfolio across years is not entirely clear. While the 2019-optimized portfolio performed well under normal conditions, its risk profile worsened during the volatility of 2020, suggesting that market dynamics can shift substantially over time. This raises the question of whether a static allocation is sufficient, or if re-optimizing the portfolio more frequently could provide better protection. In later sections, we explore this idea by testing the effectiveness of updating the portfolio on a monthly basis.

Part 3: 2019 portfolio with β =0.90 and β=0.99 (Sensitivity Analysis)



Varying the confidence level β meaningfully changed the portfolio's risk profile. With β = 0.90, the optimized portfolio achieved a lower CVaR of 0.89%, reflecting a focus on a broader portion of the loss distribution and producing a slightly less conservative portfolio. In contrast, with β = 0.99, the CVaR increased to 1.25%, as the optimization emphasized the extreme tail of the distribution, resulting in a more conservative allocation. Compared to the baseline β = 0.95 case (CVaR = 1.11%), these results illustrate the trade-off: lower β values smooth risk over more scenarios, while higher β values concentrate protection on the rarest, most severe losses.

So, how do different β values affect portfolio allocation?

2019 Portfolio β=0.9

	weight
CHTR	24.0605%
XEL	21.2977%
MDLZ	13.6573%
EXC	6.4285%
WBA	5.2900%
EBAY	4.8747%
AMGN	4.7991%
CTXS	4.7285%
ORLY	4.1987%
PEP	3.1621%

2019 Portfolio β=0.95

	weight
XEL	30.3917%
CHTR	26.4965%
AMGN	8.0766%
CTXS	6.5614%
CHKP	6.2696%
EXC	5.2155%
KHC	4.0510%
CSX	3.4312%
COST	3.0352%
EBAY	2.3486%

2019 Portfolio β=0.99

	weight
XEL	44.7035%
CHTR	24.8055%
CTXS	8.9074%
KHC	6.7269%
NTES	5.1005%
SPLK	4.6601%
AMGN	3.6309%
DXCM	0.8189%
SIRI	0.6463%

As β increases from 0.90 to 0.99, the portfolio becomes increasingly concentrated. XEL dominates across all β levels but grows from 21.3% (β=0.90) to 30.4% (β=0.95) to 44.7% (β=0.99). This suggests that as we focus on more extreme tail risks, the optimizer favors safer, utility-like stocks.

Asset Selection Changes: Different β values lead to completely different asset selections:

- β=0.90: Most diversified with 13 active positions including MDLZ (13.7%), ORLY (4.2%), and PEP (3.2%)
- β=0.95: Intermediate diversification with 10 active positions, adding defensive stocks like COST and CSX
- β=0.99: Least diversified with only 8 active positions, introducing new holdings like NTES (5.1%) and SPLK (4.7%)

Lower β (0.90) includes more consumer discretionary and cyclical stocks (ORLY, PEP, MDLZ), while higher β (0.99) shifts toward utilities (XEL dominance) and technology services, reflecting a preference for more defensive positioning against extreme scenarios.

In conclusion, higher β values force the model to be extremely conservative, concentrating in assets that perform better during tail risk events. This creates a trade-off between diversification and tail risk protection. β=0.99 offers better protection against losses but at the cost of concentration risk and potentially lower returns during normal market conditions.

Part 4: Minimizing Monthly CVaR

The goal of Part 4 problem is to implement a conservative risk management strategy for the portfolio. The traditional CvaR optimization (part 2) minimizes average tail risk across all periods (over the entire year), but this approach may still expose portfolios to severe losses during particularly adverse months. The conservative approach addresses this limitation by minimizing the maximum monthly CvaR- essentially optimizing for the worse-case scenario, which is likely higher than the average case

Objective: Minimize max{CvaR_month1, CvaR_month2,….,CvaR_month12} Subject to: Standard portfolio and return constraints.

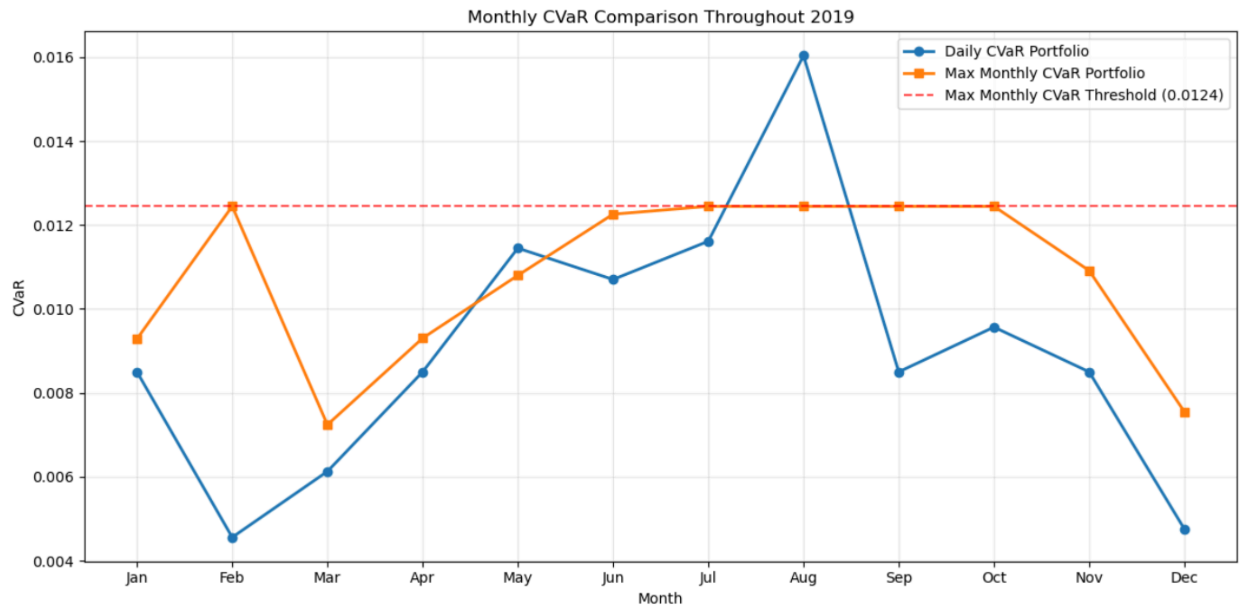
```python
# Decision variables
x = model.addMVar(n, lb=0, name="x")
alpha_m = model.addMVar(num_months, name="alpha_m") # VaR for each month
u_m_k = {} # Auxiliary variables for each month and day
Z = model.addMVar(1, name="Z") # Max CVaR

# Set objective to minimize the maximum monthly CVaR (Z)
model.setObjective(Z, sense=gp.GRB.MINIMIZE)

# Constraints for each month
for m in range(num_months):
    q_m = monthly_data[m].shape[0]
    # Define auxiliary variables for the current month
    u_m_k[m] = model.addMVar(q_m, lb=0, name=f"u_{m}")

    # The CVaR for month m must be less than or equal to Z
    cvar_m_expr = alpha_m[m] + (1 / ((1 - beta) * q_m)) * gp.quicksum(u_m_k[m])
    model.addConstr(cvar_m_expr <= Z)

    # Constraints on u_m_k
    returns_month = monthly_data[m]
    for k in range(q_m):
        model.addConstr(u_m_k[m][k] >= -gp.quicksum(returns_month.iloc[k, j] * x[j] for j in range(n)) - alpha_m[m])

# Overall portfolio constraints (apply to the whole year)
model.addConstr(gp.quicksum(x) == 1)
mean_returns_total = returns_2019.mean(axis=0)
model.addConstr(gp.quicksum(mean_returns_total[j] * x[j] for j in range(n)) >= min_return_R)
```

Python code snippet for minimizing maximum monthly CVaR

Approach: We group 2019 daily returns into monthly subsets and formulate a linear program to control risk across months. The model minimizes the maximum CVaR among all months by introducing per-month VaR and auxiliary loss variables. A portfolio weight vector is chosen such that:

1. The total weights sum to 1 (fully invested portfolio).
2. The expected annual return exceeds a required threshold R.
3. For each month, the CVaR is constrained to be below a common bound Z, which the model minimizes.

This strategy is more conservative, as it provides better protection against extreme adverse conditions in any given month. This method ended up getting a CVaR of around 1.24%. As can be seen in the provided graph, the overall monthly CVar for the conservative approach is much more stable as compared to the annual approach used in part 2 where we can see an extreme CVaR value of 1.6% in August that gets balanced out with lower CVaRs of around 0.4% and 0.6% in February and November respectively.



Part 5: Monthly Rebalancing Strategy Analysis

Static portfolio strategies, while computationally efficient, fail to adapt to evolving market conditions. The part 5 analysis implements dynamic rebalancing that recalibrates portfolio weights monthly using 12-month rolling training windows.

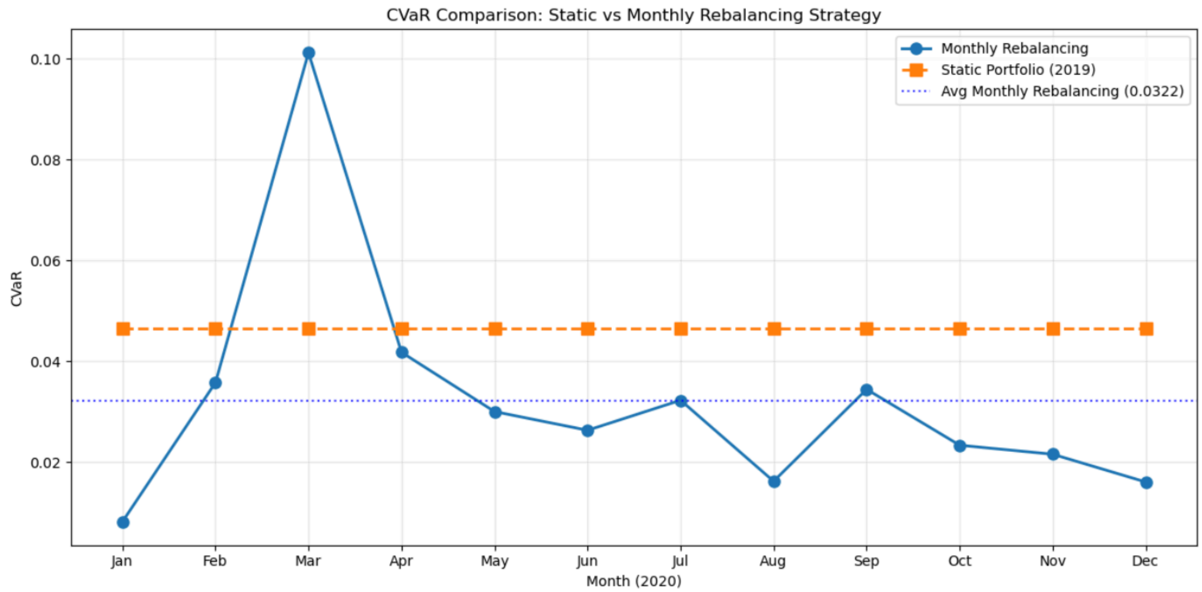
This approach addresses several critical limitations of static optimization:

Non-stationarity
Market regimes change, requiring adaptive risk models

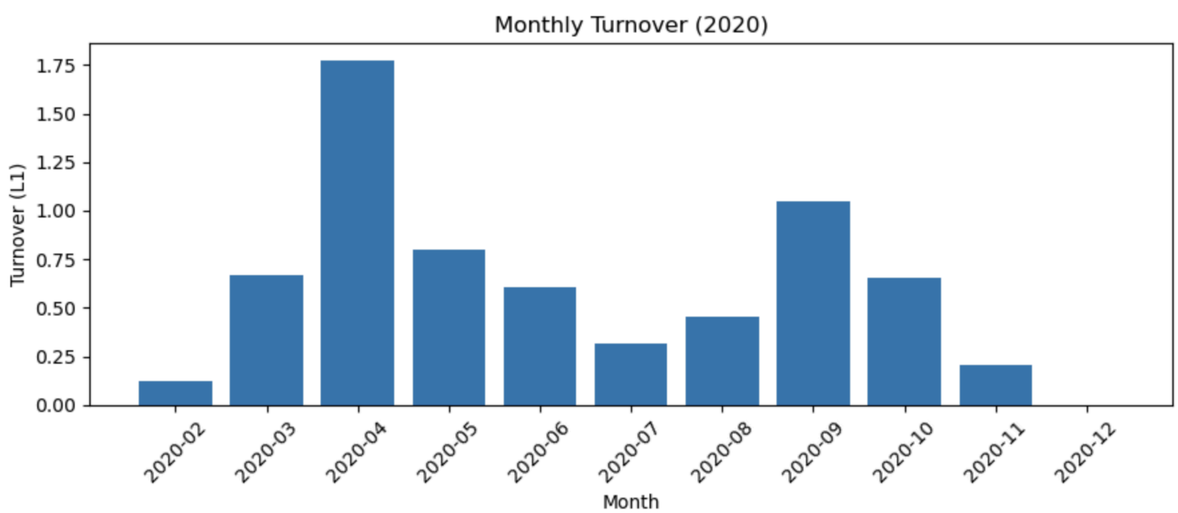
Information Decay
Older return data becomes less relevant for current optimization

Crisis Response
Dynamic rebalancing can respond to emerging market stress in real-time

Re-optimizing the portfolio each month in 2020 produced an average daily CVaR of 3.22%, compared to higher tail risk when sticking with a static allocation. The monthly approach also revealed substantial variation in risk exposure, with CVaR ranging from as low as 0.81% in stable months to as high as 10.12% during periods of severe stress. This suggests that while monthly rebalancing can reduce average risk and adapt to changing conditions, it does not fully eliminate exposure to extreme tail events. Overall, the results indicate that re-optimization provides more flexibility and generally lower downside risk than holding a fixed portfolio across the entire year.



Monthly Turnover Analysis:
The turnover chart shows that monthly reoptimization responds aggressively in stressed markets and is comparatively restrained in quieter periods. This pattern favors adaptability over allocation stability, with substantial repositioning during times of market stress. The scale of these shifts likely introduces meaningful implementation frictions from trading costs and operational complexity. The key takeaway is that monthly reoptimization improves responsiveness to changing risk but does not produce smooth or stable exposures, suiting an investor who accepts greater turnover in exchange for a portfolio that closely tracks changing market conditions.



So, the big question is- Is it worthwhile to re-optimize across months?

Looking at the numbers, monthly rebalancing performs better on average but comes with trade-offs:

The Good: The monthly rebalancing strategy achieves an average daily CVaR of 0.032, which is about 31% lower than the static portfolio's CVaR of 0.047. This means on average, you're getting better risk protection by updating your portfolio each month. The strategy also shows it can adapt - notice how CVaR drops significantly after the March 2020 crisis, suggesting the model learns from recent market conditions.

The Bad: The monthly approach is much more volatile. Your risk varies dramatically month to month (variance of 0.0005), and you get hit with extreme spikes like March 2020's CVaR of 0.101 - more than double the static portfolio's risk. This high variability makes it harder to predict and plan for risk.

So, whether it's worthwhile or not depends on your priorities. If you want lower average risk and can handle unpredictable monthly variations, monthly rebalancing wins. But if you prefer consistent, predictable risk levels and want to avoid potential extreme losses during crisis periods, the static approach might be better. The transaction costs and complexity of monthly rebalancing also need to be considered in practice.

Part 6: Portfolio Stability Assessment

We found that our monthly portfolio allocation changed by an amount greater than 5% threshold, showing that it is not stable.

Analysis of instability: The monthly rebalancing strategy produces portfolios with weight changes exceeding 5 percentage points between consecutive months. This instability likely occurs because each month's optimization uses a different 12-month rolling window of historical data, causing the model to react significantly to new information while dropping older data. Market volatility, especially during 2020 with events like the COVID-19 pandemic, would amplify these changes as the model adjusts to rapidly changing risk-return characteristics.

Proposed Solution to enforce stability - Adding Constraints to enforce portfolio stability

1. **Weight change constraints:** To enforce stability in portfolio weights over time we can add a set of constraints for each month's optimization problem.

   For a given stock *j,* let x_{jt} be its weight at time t and $x_{j(t-1)}$ be its weight from the previous month's optimization. The stability constraints would be:

   $x_{jt} - x_{j(t-1)} \leq 0.05$

$x_{j(t-1)} - x_{jt} \leq 0.05$

This can be written as $\left| x_{jt} - x_{j(t-1)} \right| \leq 0.05$. We would need to pass the previous month's optimal *x* into the next month's optimization model as a fixed parameter and add these constraints.

2. Alternative Approach: Add a penalty term to the objective function that penalizes large portfolio changes: minimize CVaR + λ $+ \sum |x_j^t - x_j^{(t-1)}|$, where λ controls the trade-off between risk minimization and stability. Higher λ values prioritize stability over CVaR minimisation, while lower values allow more portfolio changes.

This would create more stable portfolios at the potential cost of slightly higher CVaR, as the model would be constrained from making optimal but large adjustments to changing market conditions.