MIE377 - Project 2

Aakash Budhera (1004974272)
aakash.budhera@mail.utoronto.ca

Kelvin Wallace (1005314964)
kelvin.wallace@mail.utoronto.ca

Rafay Kalim (1004928741)
rafay.kalim@mail.utoronto.ca

April 18, 2021

Contents

1 Introduction

In this report, we outline the research process through which we selected and validated an automated asset management system. We tried to implement and compare each optimization method as presented in class, making some tweaks along the way to extend the ideas. As part of the analysis, we considered Sharpe Maximization, CVaR Optimization, Risk Parity, and Monte Carlo Portfolio Simulations. We compared mainly the stability, Sharpe Ratio and turnover for each method. Future returns were also modelled using two different methods, one being a stepwise regression applied to a multi-factor model. This method was extremely effective in Project 1, and is thus going to be reused for this project. Forward projections for CVaR implementations were also done via Monte Carlo Simulation to generate more robust price paths. For each of these methods, hyperparameters were optimized for the highest Sharpe Ratio for different data sets, with the turnover rate being a strictly secondary consideration.

The same design considerations from Project 1 were applied herein, with dynamic code preferred and adaptation to different economic environments. The same robust design perspective applied to project 1 was recycled with a lot of the same considerations and methods being adapted to new methods, such as Robust CVaR and Monte Carlo methods and was the leading reason for the re-use of stepwise regression and its ability to dynamic select relevant factors for each security. Additionally, other methods such as varying model based on market conditions were also considered and tested. Each of these methods had wildly varying results that all appeared to be semi-consistent across the different data sets, which led to the validation of the final model proposed herein.

2 Base Model

We are incorporating some aspects from Project 1 as a basis on which we will further develop Model 2.

2.1 Factor Model

Since we did not learn any new factor models, we will use our research from Project 1 and use Step-wise regression as the main Factor Model for this project as well. Project 1 concluded that Step-wise regression did not rely on any hyperparameters, and it was adaptable to various market conditions, and so it is attractive for this use-case as well.

2.2 Robust MVO Model

The robust MVO model was a key component of the Project 1 portfolio allocation algorithm. We concluded that the best performing confidence interval for the box uncertainty set was 90%. However, in testing we found that the same confidence interval limits were infeasible with a lookback period of 3 years (compared to the 4.5 years in project 1). As a result, we decided to lower the bounds of our confidence interval testing.

Confidence Interval	Sharpe Ratio	Turnover
0.5 (Non-Robust)	0.14624	0.44611
0.6	0.14305	0.53098
0.75	0.12943	0.83315
0.8	0.08069	1.0036
0.9	infeasible	infeasible
0.99	infeasible	infeasible

Table 1: Robust MVO Confidence Interval Test Results

2.3 Turnover Penalty

In order to reduce turnover, we included a turnover penalty in Project 1. However, we concluded that with appropriate upper bounds, the turnover penalty does not contribute much to lower the final turnover. Hence, we have chosen here to not include the turnover penalty, and focus on refining our upper bound instead.

2.4 Dynamic Target Returns

The major conclusion from our first Project was that dynamically setting the target return (i.e. by trying a variety of target returns, and picking the best one) gave us the highest returns. We a dynamic strategy in Project 2 as well, however it checks the market state to select an appropriate model that than using a brute force iteration mechanism to set a target return.

3 Optimization Techniques

3.1 Testing Defaults

For all testing done in this section, we use the following default parameters, except the one that is being changed for sensitivity analysis:

Parameter	Value
Dataset	1
Optimization Type	Risk-Parity
Factor Model Type	Step-wise regression
Test Years	3
Turnover Penalty	Off
Turnover λ	0.01
CVaR α	0.9
Monte Carlo $nPortfolios$	20000
Monte Carlo $nPaths$	20000
Lower Bound	0
Upper Bound	0.5
Robust CI	0.95

Table 2: Default Testing Parameters

3.2 Sharpe Ratio Maximization

3.2.1 Robust Sharpe Ratio Optimization

Sharpe Ratio Maximization Problem:

$$\begin{aligned} \max_x \quad & \frac{\boldsymbol{\mu}^T \boldsymbol{x} - r_f}{\sqrt{\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}}} \\ \text{s.t.} \quad & \boldsymbol{\mu}^T \boldsymbol{x} \geq R \\ & \mathbf{1}^T \boldsymbol{x} = 1 \\ & (\boldsymbol{x} \geq \mathbf{0}) \end{aligned}$$

Reformed Sharpe Ratio Maximization Problem:

$$\begin{aligned} \min_{y,\kappa} \quad & \boldsymbol{y}^T \boldsymbol{Q} \boldsymbol{y} \\ \text{s.t.} \quad & (\boldsymbol{\mu} - \bar{\boldsymbol{r}}_f)^T \boldsymbol{y} = 1 \\ & \mathbf{1}^T \boldsymbol{y} = \kappa \\ & (\boldsymbol{y} \geq \mathbf{0}) \\ & \kappa \geq 0 \end{aligned}$$

Where:

$$\begin{aligned} & \kappa = \frac{1}{(\boldsymbol{\mu} - \bar{\boldsymbol{r}}_f)^T \boldsymbol{x}} \\ & \boldsymbol{y} = \kappa \boldsymbol{x} \end{aligned}$$

While in project 1 a crude form of Sharpe ratio maximization was implemented, in this project the proper model was tested. The optimization problem was reformulated to be a quadratic programming problem that aimed to maximize the Sharpe ratio. This model was then used with various lookback periods.

Box Implementation

$$\boldsymbol{\mu}^{\boldsymbol{true}} \in M_\mu = \{\boldsymbol{\mu}^{\boldsymbol{true}} \in \mathbb{R}^n \ : |\boldsymbol{\mu}^{\boldsymbol{true}} - \mu_i| \leq \delta_i, i = 1, \dots, n\}$$

The δ_i value is decided with a confidence interval for which we want our portfolio immunized for.The final optimization problem is as follows:

Box MVO Sharpe Maximization Problem:

$$\begin{aligned} \min_{y,z} \quad & \boldsymbol{y}^T \boldsymbol{Q} \boldsymbol{y} \\ \text{s.t.} \quad & (\boldsymbol{\mu} - \bar{\boldsymbol{r}}_f)^T \boldsymbol{y} - \boldsymbol{\delta}^T \boldsymbol{z} \geq 1 \\ & \boldsymbol{z} \geq \boldsymbol{y} \\ & \boldsymbol{z} \geq -\boldsymbol{y} \\ & \mathbf{1}^T \boldsymbol{y} \geq 0 \\ & (\boldsymbol{y} \geq \mathbf{0} \end{aligned}$$

Where:

$$\begin{aligned} & \boldsymbol{y} = \kappa \boldsymbol{x} \\ & \kappa = \frac{1}{(\boldsymbol{\mu} - \bar{\boldsymbol{r}}_f)^T \boldsymbol{x}} \\ & \delta_i = \varepsilon_1 \frac{\sigma_i}{\sqrt{T}} \end{aligned}$$

Where:

$\boldsymbol{\mu}^{\boldsymbol{true}}$	-	Vector of True but Uncertain Expected Returns
$\bar{\boldsymbol{r}}_f$	-	The Risk Free Rate
δ_i	-	Maximum Error Between Actual Means and Estimates
δ_i	-	$\varepsilon_1 \frac{\sigma_i}{\sqrt{T}}$
ε_1	-	Inverse of the Two-Tailed Standard Normal CDF to the desired degree of certainty
α	-	$1 - \frac{\alpha}{2}$ is the Desired Confidence Interval

Here, the Sharpe ratio maximization problem is made robust to ensure good portfolio behaviour with uncertain expected returns. This model was used with varying confidence intervals to find what uncertainty set size would perform best. This sensitivity analysis was performed across a range of lookback periods.

Sharpe Ratio	0.5 (Non-Robust)	0.6	0.75	0.9	0.95
12mo	0.1479	0.1277	infeasible	infeasible	infeasible
24mo	0.1618	0.1579	infeasible	infeasible	infeasible
36mo	0.1043	0.1083	0.0934	infeasible	infeasible
48mo	0.1170	0.1278	0.1279	0.0613	0.0006
54mo	0.1820	0.2001	0.2244	0.1264	0.1676
Turnover	**0.5 (Non-Robust)**	**0.6**	**0.75**	**0.9**	**0.95**
12mo	1.0249	1.1233	infeasible	infeasible	infeasible
24mo	0.7724	0.8396	infeasible	infeasible	infeasible
36mo	0.6713	0.7023	0.8248	infeasible	infeasible
48mo	0.5827	0.6045	0.6753	1.0504	0.9705
54mo	0.5728	0.5663	0.6476	0.8592	0.6745

Table 3: Sensitivity Analysis of Lookback Period and Confidence Interval for Robust Sharpe Ratio Maximization

These results mirror our findings with robust MVO. After a certain confidence interval, performance steadily declines to infeasibility. There also seems to be a strange relationship between performance and lookback period, where the better performing algorithms were either very short (allowing for greater model flexibility) or the longest possible (giving the model more data to work with). Curiously, intermediate lookback periods had remarkable poor performance possibly due overcompensation inflection periods in the market. The trend with turnover was more consistent, consistently decreasing with increased lookback and consistent with confidence intervals unless the model was near infeasibility. The best performing range of parameters overall was a confidence interval of 60% to 75% with a 54 month (4.5 years) lookback period.

3.3 CVaR Optimization

CVaR Optimization:

$$
\begin{aligned}
\min_{\boldsymbol{x},z,\gamma} \quad & \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s \\
\text{s.t.} \quad & z_s \geq 0 \\
& z_s \geq f(\boldsymbol{x}, \hat{\boldsymbol{r}}_s) - \gamma \\
& s \in 1, \ldots, S \\
& \boldsymbol{x} \in \chi \\
\text{Where:} \quad & \\
& \chi = \{\boldsymbol{x} \in \mathbb{R}^n \ : \ \mathbf{1}^T\boldsymbol{x} = 1; \ldots; \boldsymbol{\mu}^T\boldsymbol{x} \geq R\}
\end{aligned}
$$

Where:

- α - Desired level of confidence for loss to exceed specified amount
- $\boldsymbol{x}$ - Vector of Portfolio Weights
- $\hat{\boldsymbol{r}}_s$ - Vector of Random Asset Returns
- $f(\boldsymbol{x}, \boldsymbol{r})$ - Loss of portfolio for a realization of random asset returns
- γ - Loss constraint
- z_s - Return for asset s based on a realized scenario

Conditional variance at risk, or CVaR, is a measure of risk involved aimed at finding the expected value of losses sustained in a portfolio past the value at risk or VaR. This tailed measure of downside risk is optimized to reduced the magnitude of extreme losses within a certain confidence interval. We first tried CVaR with historical scenarios, and then with scenarios generated through Monte-Carlo simulations.

3.3.1 Turnover Penalty

During sensitivity testing, it was observed that CVaR methods resulted in higher than expected turnover ratios. In an attempt to optimize this, a turnover penalty was added to the previous CVaR implementation. The issue with our turnover penalty from Project 1 was that it required a quadratic objective function. This is not a problem for Risk Parity or MVO, but for CVaR Optimization, which is a linear optimization problem, we decided to implement a new turnover penalty for it. To do this, a turnover penalty term, $\lambda|x - x_0|$ was added to the objective function objective. This new objective function is designed as such:

$$\gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s + \lambda|x - x_0|$$

Where λ is a constant, and x_0 is a vector of our existing portfolio. To make this into a linear program, a dummy variable $y = |x - x_0|$, was introduced to yield the objective function:

$$\gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s + \lambda y$$

With additional constraints:

$$\boldsymbol{x} - \boldsymbol{y} \leq \boldsymbol{x_0}$$

$$-\boldsymbol{x} - \boldsymbol{y} \leq -\boldsymbol{x_0}$$

Executing this yielded the following results for different alterations of the turnover penalty:

Turnover Penalty	Sharpe ratio	Average Turnover
0	0.15511	0.24431
0.001	0.13814	0.11663
0.01	0.14815	0.02768
0.1	0.14408	1.34e-16
0.5	0.14408	1.34e-16

Table 4: Turnover Penalty vs Sharpe Ratio and Average Turnover

As intended, this addition effectively limited the turnover ratio as the penalty was increased with only slight impacts to the Sharpe Ratio. Eventually when the penalty was increased to 0.5, the turnover converged to effectively 0 without a change in Sharpe Ratio. Though this step was effective in controlling turnover, its less than stellar Sharpe Ratio led us to look elsewhere for final models.

3.3.2 Robust CVaR Optimization

CVaR Optimization:

$$
\begin{aligned}
\min_{x,z,\gamma} \quad & \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s \\
\text{s.t.} \quad & z_s \geq 0 \\
& z_s \geq f(\boldsymbol{x}, \hat{\boldsymbol{r}}_s) - \gamma - \boldsymbol{\delta}_s \\
& s \in 1, \ldots, S \\
& \boldsymbol{x} \in \chi
\end{aligned}
$$

Where:

$$
\begin{aligned}
& \chi = \{\boldsymbol{x} \in \mathbb{R}^n \ : \ \mathbf{1}^T\boldsymbol{x} = 1; \ldots; \boldsymbol{\mu}^T\boldsymbol{x} \geq R\} \\
& \boldsymbol{\delta}_i = \varepsilon_i \frac{\sigma_i}{\sqrt{T}}
\end{aligned}
$$

Robust CVaR takes the nominal historical CVaR level and applies an uncertainty set around the returns vector. The CVaR is then optimized about this uncertainty set, the size of which is governed by the parameter δ. The model was run with a range of uncertainty parameters to assess how it performed, holding α constant at 0.6.

Confidence Interval	Sharpe Ratio	Turnover
0.5 (Non-Robust)	0.15511	0.24431
0.6	0.15548	0.24741
0.75	0.12799	0.30511
0.8	0.13241	0.36258
0.9	0.12549	0.42566
0.99	infeasible	infeasible

Table 5: Robust CVaR with Box Uncertainty Set Confidence Interval Test Results

As with robust MVO, there is a decline in performance corresponding to a larger confidence interval, an expected result. However, to maintain some level of robustnes, a small confidence interval should still be used with robust CVaR.

3.3.3 Historical CVaR

This CVaR optimization model uses historical returns to compute CVaR. It uses all data available (last 5 years) to compute CVaR. A limitation of this technique is that if the last 5 years of data do not have a significant downturn, the model will not accommodate for a potential bear environment, and can hence perform rather poorly. One key parameter for the historical CVaR optimizer is our Confidence Interval, α, on which to stop losses. We tried a variety of values and report the results here:

α	Sharpe Ratio 1	Turnover 1	Sharpe Ratio 3	Turnover 3
0.6	0.13858	0.28172	0.13236	0.2938
0.75	0.15022	0.27947	0.15225	0.30076
0.9	0.15215	0.26986	0.13597	0.26805
0.95	0.1952	0.33635	0.19528	0.20424
0.99	0.21959	0.29956	0.18743	0.23372

Table 6: CVaR Confidence Interval Test Results Across Datasets 1 and 3

Table 6 summarizes the results from varying the Confidence Interval on historical CVaR. Sharpe Ratio and Turnover 1 correspond to results with data from the First Dataset file, while Sharpe Ratio and Turnover 3 correspond to results with data from the Third Dataset file. We see an immediate benefit of increasing the confidence interval. These results agrees with theory. A larger CVaR means that we are more protected against downturns, and this explains why we get better results when there is a downturn. However, it also leads to a more risk-averse portfolio, and so it can also lead to lower Sharpe Ratios if it is not targeting high returns. Hence, a 95% Confidence Interval may be ideal as it gives the highest incremental improvement over the last interval, and also does not lead to portfolios that are too risk-averse.

3.3.4 Monte Carlo CVaR Simulations

For this method, scenarios were randomly generated using a Monte Carlo simulation then fed into the previous CVaR Optimization method to obtain the optimal portfolio. $nPaths$ Scenarios were generated via a under the following parameters:

Let:

nPaths	be the number of scenarios generated
n	be the number of assets in the portfolio
$\varepsilon \in \mathbb{R}^n$	be a vector of n $\mathcal{N}(0,1)$ independent variables
$\boldsymbol{\rho} \in \mathbb{R}^{nxn}$	be the correlation matrix of assets
$\boldsymbol{L}$	be $\in \mathbb{R}^{nxn}$ and $\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{\rho}$
$\boldsymbol{\xi}$	be $\boldsymbol{L}\boldsymbol{\varepsilon}$

With prices were randomly generated as:

$$S^i_{t+1} = S^i_t \exp\left[(\mu_i - \frac{\sigma_i^2}{2})dt + \sigma^i \sqrt{dt}\ \xi^i_t\right]$$

A natural question to ask at this point is how high $nPaths$ should be. Hence, we conducted a test by varying this parameter and examining how results differ. Table 7 summarizes our results.

$nPaths$	**Sharpe Ratio**	**Turnover**
10	0.14891	0.25553
50	0.15099	0.29009
100	0.16868	0.33306
250	0.18494	0.33356
500	0.18455	0.31198
5000	0.18455	0.31198

Table 7: Monte Carlo CVaR Number of Scenarios Test Results

There are some interesting trends to be analyzed here. First, we note that increasing the number of simulations does indeed improve the Sharpe Ratio of our algorithm. However, it is interesting to see that the algorithm does not show much marginal improvement when increasing the number of simulations from 100 to 5000. While not included in this report,

we tested this on other datasets and saw a similar "convergence" at fairly low number of simulations.

This was concerning for us, as intuitively we thought that increasing the number of simulations should affect performance. Hence, we wondered if this algorithm was converging at a low number of simulations. If it was, we could take advantage of this by adding more randomness and hence converging to a better solution.

We proceeded to test this hypothesis. We added more randomness in this way: in each simulation, we randomly choose which of these three categories it will be a part of:

1. Bull Market - Here, the μ vector (when randomly generating the prices) is perturbed by +5% annualized.
2. Bear Market - Here, the μ vector is perturbed by -5% annualized.
3. Stagnant Market - Here, the μ vector is left as is.

By performing this analysis, we see the following results summarized in Table 8.

$nPaths$	**Sharpe Ratio**	**Turnover**
10	0.14862	0.269
50	0.15397	0.29796
100	0.16868	0.31098
250	0.16408	0.33356
500	0.16774	0.32093
5000	0.16813	0.34164

Table 8: Monte Carlo CVaR Number of Scenarios with Random Perturbation Test Results

The results show that the random perturbations have the effect of lowering our Sharpe Ratio, while still having the same issue of early convergence. To conclude, we are not completely sure why this early convergence is happening. In the future, it may be a good idea to explore this idea further in order to take advantage of this early convergence by exploiting the fact that we do not have as much of a time constraint due to it. We are choosing 500 simulations as our baseline going forward, as increasing more than that decreases speed but does not give any improvement.

3.4 Risk Parity Portfolio Optimization

Risk Parity:

$$\begin{aligned} \min_x \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}(x_i(\boldsymbol{Q}\boldsymbol{x})_i - x_j(\boldsymbol{Q}\boldsymbol{x})_j)^2 \\ \text{s.t.} \quad & \mathbf{1}^T\boldsymbol{x} = 1 \\ & \boldsymbol{x} \geq \mathbf{0} \end{aligned}$$

The risk parity model seeks to equalize the risk contribution of assets in a portfolio's overall variance. This creates a portfolio where risk is truly diversified and unreliable return estimates do not need to be used. The general model takes no inputs other than the covariance

matrix, which is extremely helpful in high volatility environments. Consequently, various methods to calculate this matrix can be used to vary the model outputs such as the lookback period or choosing implied over historic volatility. One issue is that the formulation is not robust when it comes to altering parameters and adding additional constraints. This is apparent when applying transformations to x to make the solution easily solve-able. The necessary transformation results in the substitute variable not allowing for an upper bound, as we we could not find a way to translate a the constraint into an optimization function. This leave the only parameters we can realistically alter are lookback period and how the covariance matrix calculated.

3.4.1 Risk Parity Lookback Period

One of the few parameters out implementation of Risk Parity could alter was the lockback period, or how much past data was considered when calculating the covariance matrix. Data for changing lookback periods (in years) versus Sharpe Ratio and turnover ratio:

Lookback Period	Sharpe Ratio	Turnover Ratio
1	0.19472	0.29762
1.5	0.16938	0.25821
2	0.16881	0.2204
3	0.16676	0.18001
4	0.17045	0.16252
4.5	0.17357	0.15346

Table 9: Lookback Period vs Sharpe Ratio and Turnover Ratio

3.4.2 Risk Parity Long/Short Limits

By design, the convex formulation of risk parity does not allow short selling due to its $\sum_{i=1}^{n} ln(\boldsymbol{x}_i)$ term. Hence, we did not get the opportunity to implement Long/Short Limits. However, in the future we wanted to test if it would be possible to reformulate the problem as follows:

$$f(y) = \frac{1}{2} y^T Q y - c \sum_{i=1}^{n} ln(|y_i|)$$

Note that here we apply an absolute value sign on y_i, and so this allows us to short sell. Unfortunately, we did not have time to implement this. Additionally, due to the transformation of $\boldsymbol{x}$, we did not find a way to introduce upper bound constraints on y that translated to the proper upper bound on x when the portfolio is normalized. As noted above, the Risk Parity condition of the objective function disallows short positions, as a negative number will not be defined. If there was more time to perform analysis, we would implement the above proposed objective function to implement a long/short risk parity portfolio.

3.5 Monte Carlo Optimization Techniques

In this section, we consider some stand-alone Monte Carlo Optimization Techniques. These techniques revolve around one central idea: generating a lot of different scenarios, and then picking results from the one that gives us the best results.

3.5.1 Random Portfolios

The most simplest usage of Monte Carlo involved generating $nPortfolios$ random portfolios, and using an Ex-Ante Sharpe Ratio to pick the portfolio that gave us the best results. This, while sounding rather naive, did give us some good results as we will discuss soon. To formalize this idea, the following algorithm was used:

for *i in range 1 to nPortfolio* **do**
  Generate vector $\boldsymbol{x}^i \in \mathbb{R}^n$ such that $\boldsymbol{x}^i_j \sim \mathcal{U}(0,1)$
  Normalize vector: $\boldsymbol{x}^i_j = \frac{\boldsymbol{x}^i_j}{\sum_{j=1}^{n} \boldsymbol{x}^i_j}$ such that $\mathbf{1}^T \boldsymbol{x}^i = 1$
  Find $SR_i = \frac{\boldsymbol{\mu}^T \boldsymbol{x}^i}{\boldsymbol{x}^{iT} Q \boldsymbol{x}^i}$
  if $SR_i > SR_{best}$ **then**
    $SR_{best} = SR_i$
    $\boldsymbol{x}_{best} = \boldsymbol{x}^i$
  end
end
return $\boldsymbol{x}_{best}$

Where:

nPortfolio	is the number of randomly generated portfolios
$\boldsymbol{x}^i \in \mathbb{R}^n$	is the vector of randomly generated asset weights of iteration i
$\mathcal{U}(0,1)$	is a uniform random distribution between 0 and 1
Q	is the asset covariance matrix
SR_i	is the estimated Sharpe Ratio of iteration i

A natural parameter to tune in this case was $nPortfolios$, i.e. the number of portfolios we generate. The results for this are included in Table 10. Note that each value for $nPortfolios$ was run 10 times, and the average is displayed here.

$nPortfolios$	**Average Sharpe Ratio**	**Std. Dev. of Sharpe Ratios**	**Average Turnover**
1000	0.2234	0.13	0.68737
2500	0.18171	0.15	0.65281
5000	0.21302	0.05	0.65057
10000	0.1758	0.09	0.61026
20000	0.19706	0.03	0.61897

Table 10: Randomly Generated Portfolios Average Test Results

Looking at the results, it is hard to find a discernible trend in the Sharpe Ratio by increasing the number of portfolios we generate. However, we do note that the turnover decreases.

Additionally, looking at the standard deviation of the average Sharpe ratio, we can see that there is a downward trend in how much the values deviate from the mean when increasing the number of portfolios. While it does go down, a 6% deviation from the mean for 20,000

portfolios is still quite high. Due to this uncertainty in the Sharpe Ratio, we decided not to pursue this model further.

3.5.2 Robust Monte Carlo Projected Return Weighted Portfolio (RMCPRWP)

This is another naive Monte Carlo approach that we wanted to try. This algorithm is simple as well, but depends on Monte Carlo projections for each stock. It uses the following equation, referenced and explained previously to project the pricing path of the stocks:

$$S_{t+1}^i = S_t^i \exp\left[(\mu_i - \frac{\sigma_i^2}{2})dt + \sigma_i\sqrt{dt}\ \xi_t^i\right]$$

It assumes the initial stock price is \$1. This way, the final price less 1 can be thought of as the return over the 6 month period. After we have the projected returns, the portfolio is constructed by normalizing the return vector. Hence, each asset is weighed by its normalized projected return over the next 6 months.

$$w_i = \frac{\mu_i}{\sum_{i=1}^n \mu_i}$$

Where w_i is the weight of asset i and μ_i is i-th element of the vector of projected returns.

We kept in mind the results from Monte Carlo CVaR. Recall that increasing the number of portfolios generated did not significantly impact the performance of the portfolio due to early convergence, and so we perturbed the μ vector to increase randomness. Here, we implemented the same random perturbation. This time, it randomly adds a perturbation of a scalar to all returns, in the range (−0.1/12, 0.1/12) in order to represent a +10% or -10% perturbation annualized. This, hopefully, will make the optimization more robust as it will account for a more bearish and a more bullish environment as well. We will call this "Robust" Monte Carlo Projected Return Weighted Portfolio (RMCPRWP). Now, we will perform a sensitivity analysis on $nPaths$. Here, we are interested in not just the Sharpe Ratio and turnover, but also the consistency with which we get the mean. To that end, we ran 10 trials for each value of $nPaths$, and Table 11 has the mean and standard deviation.

$nPaths$	**Average Sharpe Ratio**	**Std. Dev. of Sharpe Ratios**	**Average Turnover**
10	0.19567	0.03	0.35182
50	0.19702	0.04	0.35324
500	0.20519	0.002	0.35542
1000	0.19788	0.001	0.33252
20000	0.19677	0.0003	0.32654

Table 11: Average Results from Varying $nPaths$

As we can see from Table 11, all values of $nPaths$ gives us around the same Average Sharpe Ratio. However, once again we note that the standard deviation of the Sharpe Ratio decreases dramatically as we increase the number of scenarios. Hence, for stability purposes, we should choose the highest possible value for $nPaths$, especially since increasing it to 20,000 did not increase the computation time dramatically.

Since this model was implemented in the final Project 2 algorithm, it may have been beneficial to test various long/short constraints. However, due to the current formulation of the model, there is no provision for negative weights and thus such a test would be extraneous. Moreover, for consistency with using the risk parity algorithm alongside this model, the use of long/short portfolios has been disallowed.

3.6 Model Selection Methodology

Now that we have the optimal hyperparameters for each model, we can fairly compare the best version of each model against each other to choose the best one. In this section, we test each model against each other and rank them based on their properties.

3.6.1 Dynamic Model Selection

We noticed, when testing the models individually, that each model performed differently in each dataset. This led to us experimenting on how each model performs in a bull vs. bear environment. To do this, we manually classified the following time data as bull and bear, as summarized in Table 12.

Time Range	Classification
01/01/2002 - 31/12/2002	Bear
1/1/2000 - 31/12/2000	Bear
1/1/1992 - 31/12/1993	Bull
01/01/2003 - 31/12/2004	Bull

Table 12: Manual Classification of Time Periods

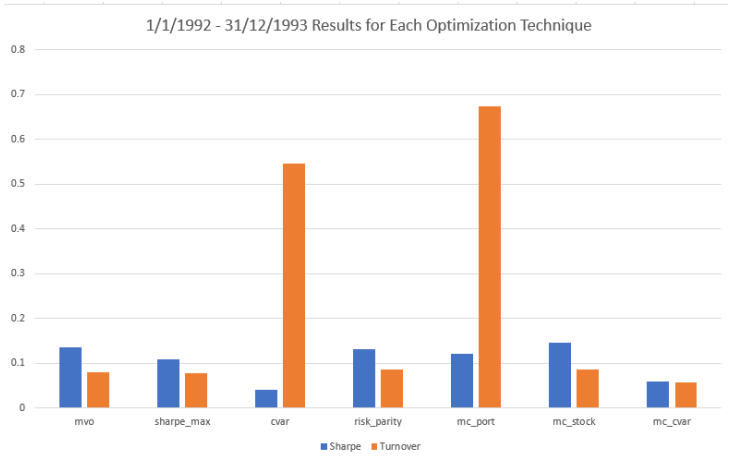
This was done by our own knowledge of the financial markets in this time, and also by looking at the market returns during these times.

Once we had the classifications of these time periods, we individually tested each model in each of these time periods. Figures 1c and 1d show the results of the bear environments, and figured 1a and 1b show the results from the bull environments.

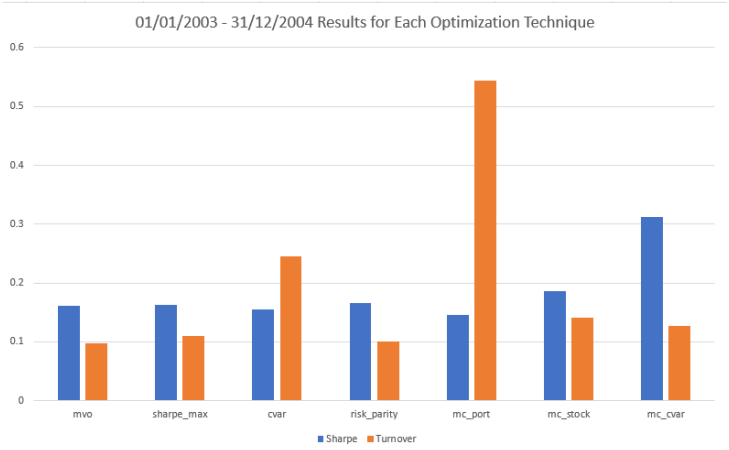
From the graphs, it is clear to see that risk parity gives us consistent returns throughout both market environments. This is not the case with other models, such as Monte Carlo CVaR, which is inconsistent between 2002 and 2000 for example. Risk parity also gives us better returns than MVO, and so is the ideal candidate in all scenarios. However, the method that stands out is the "Robust" Monte Carlo Projected Return Weighted Portfolio (RMCPRWP). In bull environments, it consistently has high Sharpe ratios and low turnovers due to its aggressive nature. Hence, our conclusion from this is that we want to choose the model to use based on market conditions. In a bear market, we want to choose a conservative Risk Parity model, and in a bull environment, we want to choose a more aggressive RMCPRWP model. Now, the problem is to identify when we are in a bull or bear environment.

3.7 Dynamically Identifying Market Environment

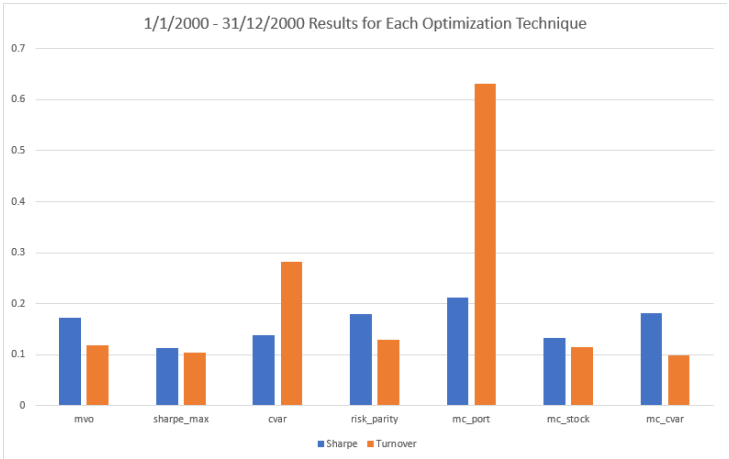
The following algorithm was used to identify the state of the market in a simple classification between bull and bear:



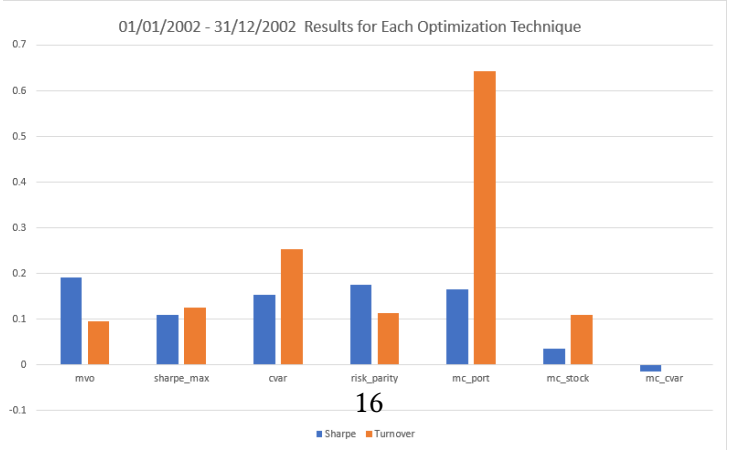
(a) Bull Environment From 1992 to 1993



(b) Bull Environment From 2003 to 2004



(c) Bear Environment From January 2000 to December 2000



(d) Bear Environment From January 2002 to December 2002

Data:

Generates a vector of 12 even spaced points to weight returns

$\boldsymbol{W} = linspace(1, 2, 12)$

Generates a vector of market returns from twelve months

$\boldsymbol{Rets} = (\boldsymbol{mktRet}(end - 11 : end, 1) + 1)^T$

Weight the geometric means based on how recently the return was observed

$\boldsymbol{\mu}_w = \left(\prod_{i=1}^{12} \boldsymbol{Rets}_i^{W_i}\right)^{\frac{1}{\Sigma W}}$

if $\boldsymbol{\mu}_w > (\prod_{i=1}^{n} \boldsymbol{mktRet}(i : i + 1, 1) + 1)^{\frac{1}{n}}$ **then**

Bull Market: weighted mean greater than mean of entire 54 month sample period:

ModelType = "RMCPRWP"

else

Market is in a bear state:

ModelType = "Risk Parity"

end

Since it is impossible to know beforehand what the future state of the market will be, our model uses data based heuristics to map possible bull or bear markets to the appropriate model. This heuristic is a weighted moving geometric average taking the past twelve months of market return, with the most recent period being weighted double compared to the furthest period. If this rolling weighted 12-month average falls below the geometric average of the market return over the entire sample period, a bear state is assigned and the appropriate model (risk parity) is assigned. Conversely, if the 12-month average is higher than the overall average, a bull state is assigned and a riskier, more aggressive model (RMCPRWP) is used.

4 Conclusion

4.1 Final Model Selection and Parameters

Our final model model combined risk parity with a robust Monte Carlo projected return weighted portfolio (RMCPRWP), dynamically selected based on the bullishness or bearishness of the market. Asset returns were estimated using a stepwise regression process. This combination generated a stable and effective portfolio selection algorithm which resulted in performance that consistently beat the market with an acceptable level of risk across all three market data scenarios. Here, we summarize the our final proposal. This model aims to solve the following minimization problems:

A bear market is defined as:

$$\boldsymbol{\mu}_w < \left(\prod_{i=1}^{n} \boldsymbol{mktRet}(i \,:\, i+1, 1) + 1 \right)^{\frac{1}{n}}$$

Which means the 12-month Exponential Moving Average is less than the 54-month Exponential Moving Average. In a bear market, Risk Parity is used to find an optimal portfolio:

Risk Parity:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i(\boldsymbol{Qx})_i - x_j(\boldsymbol{Qx})_j)^2 \\ \text{s.t.} \quad & \mathbf{1}^T \boldsymbol{x} = 1 \\ & \boldsymbol{x} \geq \mathbf{0} \end{aligned}$$

In the bull case (i.e. not a bear market), pricing paths for each stock are generated using a Monte Carlo simulation using the following equation:

$$S_{t+1}^i = S_t^i \exp\left[(\mu_i - \frac{\sigma_i^2}{2}) dt + \sigma^i \sqrt{dt}\, \xi_t^i \right]$$

Using the average projected returns of the randomly generated paths of each asset, individual portfolios weights were calculated using:

$$w_i = \frac{\mu_i}{\sum_{i=1}^{n} \mu_i}$$

The following parameters were used:

Parameter	Value
Optimization Type	Bull/Bear Signal
Bull Case	"RMCPRWP"
Bear Case	Risk Parity
Factor Model Type	Stepwise Regression
Sample Years	4.5
Monte Carlo $nPaths$	20000
Lower Bound	0
Upper Bound	1
Turnover Penalty	None

Table 13: Final Parameters

4.2 Results

Our final model model was validated on all three available datasets and compared against the baseline MVO optimization with OLS regression and our project 1 information ratio maximization algorithm (with normalized portfolios, a required fix to stay within project parameters). The algorithms give the following Sharpe ratios and turnovers across the three market scenarios:

Sharpe Ratio	MVO/OLS	Project 1	Project 2
1	0.21155	0.21359	0.23367
2	0.068965	0.11987	0.16475
3	0.18049	0.21005	0.21143

Turnover	MVO/OLS	Project 1	Project 2
1	0.4218	0.48716	0.34557
2	0.44127	0.6107	0.36228
3	0.45516	0.6997	0.36459

Table 14: Sharpe Ratio and Turnover Results for Baseline MVO/OLS, Project 1, and Project 2 Algorithms Across Three Market Scenario Datasets

As we can see, our project 2 algorithm shows considerable improvement over the baseline as well as our work from the last project. In every respect, whether it was Sharpe ratio or turnover, the new algorithm had a consistently better result over all three scenarios. On runtime, our model was much faster (approximately 5x) than the project 1 algorithm but slightly slower (about 2x) than the baseline, an expected result since the looping through 250 iterations of MVO per rebalancing period was eliminated. Dynamically adapting a model according to market state is a much more efficient and effective result than brute force iteration to find optimal portfolios.

The project 2 algorithm also promotes better diversification, while maintaining a consistent level of return. Compared to past algorithms, more assets are being invested in due to the risk parity optimization, while returns are still aggressively targeted with the RMCPRWP model. This tradeoff results in a favourable asset mix that provides downside protection alongside market beating performance. The following figures show the portfolio weight evolution alongside portfolio returns for the first dataset:

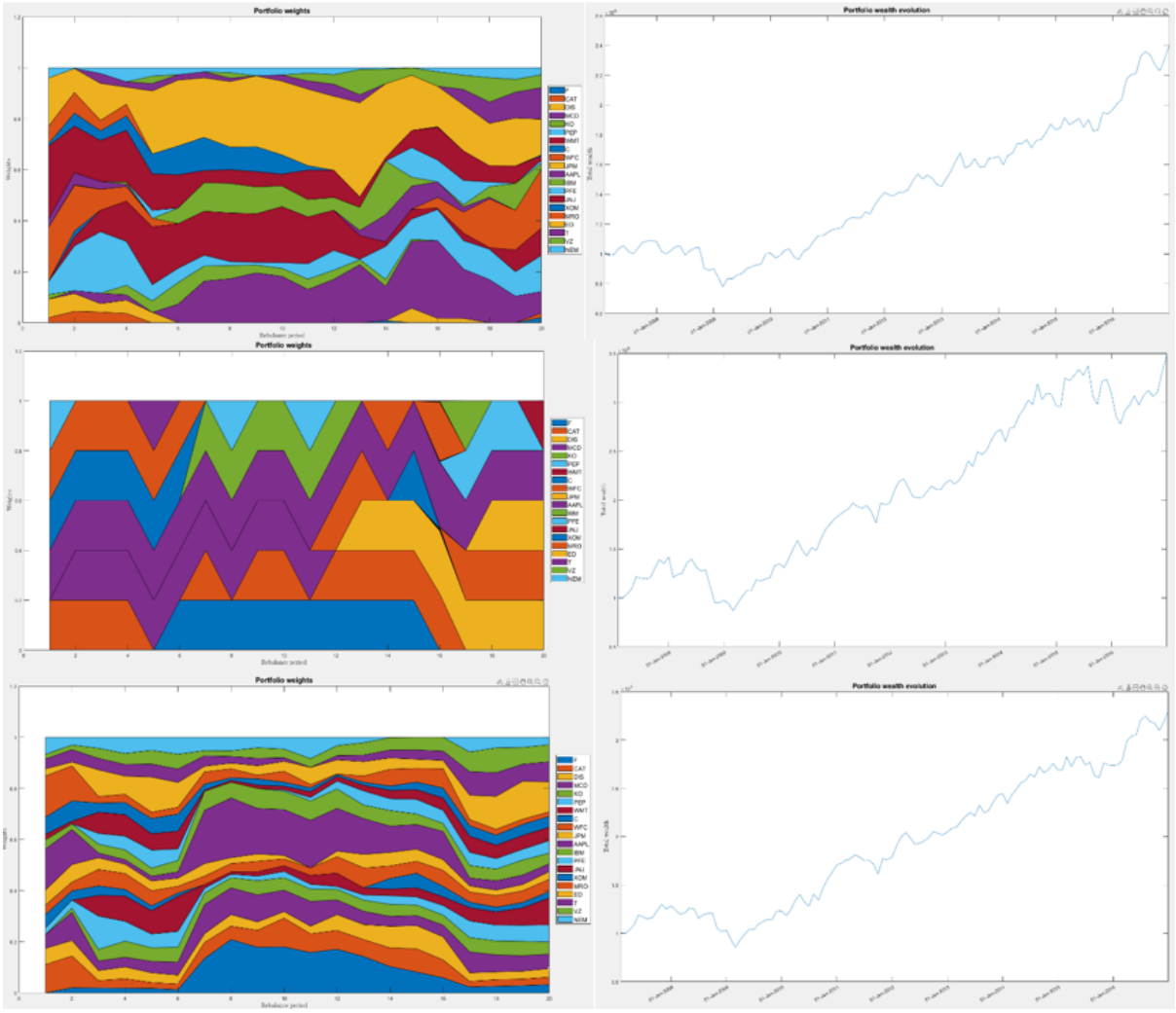


Figure 2: Analysis of Portfolio Weight Evolution and Returns for Baseline MVO/OLS, Project 1, and Project 2 Algorithms on Dataset 1

As seen above, the project 2 algorithm maintains the high returns of project 1 at around $350 thousands compared to the baseline at $240 thousand, representing a 13% annualized return compared to 9%. This is in conjunction with a high level of diversification, which can be seen visually in Figure 2. In contrast with the clunky allocations of project 1 or the concentrated positions of the baseline model, the project 2 model shows a well balanced mix of assets.

4.3 Future Considerations

There are a number of drawbacks to our model. First, our RMCPRWP model does not account for variance at all. It only projects returns into the future 6 months, and uses this to construct the portfolio. It may be a good idea to have some sort of variance-control in our final model.

Secondly, our identification of market environments depends on the 12-month moving average of the market returns. While this seemed to work for our use case, a better identification may involve a logistic (or other binary classification algorithm) regression on many factors rather than just the market return.

Next, we may do further analysis on different market states than just bull or bear, and see if a specific model works better in each one. For example, we can define extreme bull or bear environments, such as recessions (where we would want to invest in the lowest risk assets).

A last improvement which was not possible with the data provided would be to label each asset, and classify its sector or asset type (such as bonds, ETFs, equities, etc.). This would allow us to easily group stocks, limit investments in certain sectors, and provide low risk assets in bear environments.