Portfolio Optimization Applied in Stock Trading Strategies

Guillaume Eymery
Email : guillaume.eymery@ensam.eu

ABSTRACT

Abstract - This paper will focus on the application of portfolio optimization techniques within stock trading strategies. Since 1900 and Pierre Bachelier's "Theory of Speculation"[i]*, the quantitative aspect of finance has taken a predominant role for any investor. The development of Computational Science made it easier to analyze large amounts of financial data and to implement sophisticated mathematical models to be profitable.*

The primary objective of this study is to explore the effectiveness of portfolio optimization methodologies in enhancing long-term trading strategies across various stock markets. By leveraging historical market data and employing optimization algorithms, we aim to identify optimal portfolio allocations that can potentially outperform traditional investment approaches.

Portfolio Optimization in Quantitative and Computational Finance has already been theorized, notably by Michael J. Best in "Portfolio Optimization"[ii]*, but this study aims to confront the models with the reality, based on a new trading hypothesis, and to maximize profit by implementing efficient hyperparameters to the models.*

Keywords: *Portfolio optimization, Trading strategies, Computational finance, Financial markets, Optimization algorithms*

Disclaimer: The information presented in this paper is provided for educational purposes only and should not be construed as financial advice. Readers are advised to consult with a qualified financial advisor or investment professional before making any investment decisions. The author is not liable for any financial losses or damages arising from the use of the information presented herein.

I. INTRODUCTION

In investment management, portfolio optimization involves selecting the optimal combination of assets to achieve specific objectives, such as maximizing returns while minimizing risk. This concept is widely utilized in diverse fields such as Hedge Funds, Private Equity, Risk Management, Quantitative Finance, as well as personal investments. Combining investing strategies in stock trading with Python Machine Learning can sometimes provide positive results that may outperform the market in the long term. Furthermore, it is not always necessary to possess high computational power or knowledge to attain these results. Basic computational knowledge of popular and accessible Python libraries such as pandas, matplotlib, SciPy, or yfinance (Yahoo Finance) are the only prerequisites.

Contents

II. THEORETICAL FOUNDATIONS

A. The Sharpe Ratio

Calculating the Sharpe Ratio is a way to evaluate the risk-adjusted return of an investment or a trading strategy. The Sharpe Ratio is calculated by subtracting the risk-free rate of return from the expected return of the investment or strategy and then dividing this difference by the standard deviation of the investment's returns. Mathematically, it can be expressed as:

$$S_R = \frac{R_p - R_f}{\sigma_p}$$

Where:

S_r : is the Sharpe Ratio.

R_p : is the expected return of the investment or strategy.

R_f : is the risk-free rate of return, in this case, the yield of a government bond (Federal Reserve Bank of St Louis).

σ_p : is the standard deviation of the investment's returns, which measures its volatility or risk.

The Sharpe Ratio is the value that will be maximized in this paper. Since the optimization algorithms that are used in this paper are based on the gradient descent method, thus we will be minimizing $-R_p$ [iii].

The Sharpe Ratio has often been considered as a tool to analyze the effectiveness of an investment. In this article we will be utilizing this ratio as a decision making tool.

B. Drawbacks as fund selection criteria

The Sharpe Ratio is a simple way to calculate the return on an investment, however, it has several limitations and considerations:

- It assumes that returns follow a normal distribution. However, in reality, financial markets can exhibit non-normal behavior which can affect the reliability of the Sharpe Ratio (it badly reacts to Ponzi Schemes)[iv].
- It has often been challenged concerning its appropriateness as a fund performance measure during market downturns[v].

C. Mathematical overview of the optimization process

The portfolio in this study will be entirely composed of n stocks. Each of them will be associated with a weight. They will represent the variables of the optimization process that can be expressed in a vector W, where:

- $\forall k \in [1, n], 0 \leq w_k \leq 1$
- $\sum_{k=1}^{n} w_k = 1$

$$W = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n-1} \\ w_n \end{pmatrix}$$

The function that describes the portfolio's return based on the Sharpe Ratio is:

$$S_{RP}\ (w_1, \dots, w_n) = \frac{\sum_{k=1}^{n} w_k R_k - R_f}{\sigma_p}$$

Where:

R_k is the return for the specific stock k over T.

σ_p is the portfolio volatility.

In the case of this study, the volatility of a series will be defined as the square root of its variance. The covariance matrix has an important use as it captures the pairwise relationships between the returns of different assets in the portfolio. The covariance matrix allows investors to assess how the returns of one asset are correlated with the returns of other assets in the portfolio. Assets with low or negative correlations can help diversify risk, as they may not move in tandem with each other. By including assets with different correlation patterns, investors can potentially reduce the overall risk of the portfolio without sacrificing returns.

Thus:

$$\sigma_p = \sqrt{W^T . cov_R . W}$$

This leads to the vectorized equation:

$$S_{RP}\ (W) = \frac{W^T . R - R_f}{\sqrt{W^T . cov_R . W}}$$

Where: $R = (R_1, \dots, R_n)^T$

Let $NS_{RP}(W) = -S_{RP}(W)$ be the function to minimize and $W^* = \min_{W} NS_{RP}\ (W)$. Let us note that the returns are also a function of time, thus NS_{RP} is also a function of time, the chronological aspect will be studied in the next part.

The optimization problem has constraints that restrict the feasible space of W. These constraints can be expressed as equality or inequality constraints:

- Equality constraint : $g(W) = 0$, and $g(W) = W^T . b - 1$, where $b = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$
- Inequality constraints : $h_1(W) \leq 0$, and $h_1(W) = W - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$

Using an iterative optimization algorithm such as the Sequential Least Squares Programming Method (SLSQP) appears to be effective when optimizing complex nonlinear functions subject to both equality and inequality constraints.

The initialization vector W^0 is set as : $W^0 = \begin{pmatrix} \frac{1}{n} \\ \vdots \\ \frac{1}{n} \end{pmatrix}$

Built-in optimizing functions are already implemented in the open source Python Library SciPy[vi], under the class scipy.optimize and provide very satisfactory results.

Let us note that additional bounds can be computed, such as $w_k < 0.4$, that requires the shareholder to possess under 40% of any stock and prevents solutions such as $W^* = (0,0, \dots ,1, \dots ,0,0)^T$. This fairly reduces the amount of risk associated with the trading strategy.

D. Evaluating the stock returns for the optimization process

The stock returns represent the main data for this study, thus they should be evaluated carefully. While arithmetic returns are simple to calculate and provide a clear picture of the average return over a given period, they can be misleading when understanding the impact of compounding returns and unpractical when working with different periods.

$$ArithmeticReturn = \frac{ClosePrice - OpenPrice}{OpenPrice}$$

In this case, logarithm returns are more interesting to work with because they express the proportional change in value over time and are unaffected by the magnitude of the initial investment. Logarithmic returns are also commonly used in finance because of their additive properties.

$$LogarithmicReturn = \log \left(\frac{ClosePrice}{OpenPrice}\right)$$

In this case, let T be a period:

$$\forall k \in [1, n], R_k(t = T) = \log \left(\frac{ClosePrice(t = T)}{OpenPrice(t = T)}\right)$$

The return over n periods T can be expressed by additivity:

$$R_k(t = nT) = \sum_{i=1}^{n} \log \left(\frac{ClosePrice(t = iT)}{OpenPrice(t = iT)}\right)$$

Furthermore: $OpenPrice(t = iT) = ClosePrice(t = (i - 1)T),\ i : trading\ day$

Finally : $R_k(t = nT) = \log \left(\frac{ClosePrice(t=nT)}{ClosePrice(t=0)}\right)$

The stock returns will be based on the *Adjusted Close* values as they are not affected by stock splits or dividends. The commission associated with the Buy/Sell of a stock is neglected. Potential transaction fees should be covered by the dividends (that are not accounted as well). Furthermore, modern trading applications offer low trading fees for personal transactions.

This ends the problem formulation part. The next chapter will focus on the hypothesis that will guide the Buy/Sell strategy.

III. FINANCIAL HYPOTHESIS

A. About the French CAC 40

The French stock market index CAC 40 [vii]represents a selection of the 40 largest publicly traded companies in France, based on market capitalization. The CAC 40 is one of the most widely followed indicators of the French stock market's performance and is often used as a gauge for the overall health of the French economy. It includes companies from various sectors, including finance, industry, consumer goods, and services.

The advantages of buying and selling CAC 40 stocks are that the companies enlisted are generally well-established, reputable, and widely followed by investors and analysts, providing a level of stability and liquidity in the market. Therefore, the stock prices are more likely to follow a normal distribution and the volatility is quite low, which favors the usage of the Sharpe Ratio for portfolio optimization.

B. Initial computer test and analysis

The code is initially run without any trading strategy. It optimizes the Sharpe Ratio for every semester and returns the best stocks associated with their weights, as well as the returns for this semester as well as the volatility and the Sharp Ratio. An example of this code can be found in the Appendix. Here are the results with the program starting date in 2010 and ending date in 2024:

```
                Result Optimization for the semester 2023-06-16
Stocks and weights:
Bouygues: 0.1204
Crédit agricole: 0.0464
Engie: 0.2347
Publicis Groupe: 0.0838
Stellantis: 0.3594
TotalEnergies: 0.1553

Returns 0.204   Volatility 0.095

Optimal Sharpe Ratio 1.698
```

Figure 1 Initial Test Optimal Results for 06/2023

As expected, the results are excellent: the Sharpe Ratio is high, the returns are up to 20% and the volatility remains below 10%. The minimization function has perfectly converged as it guarantees important returns and small risks.

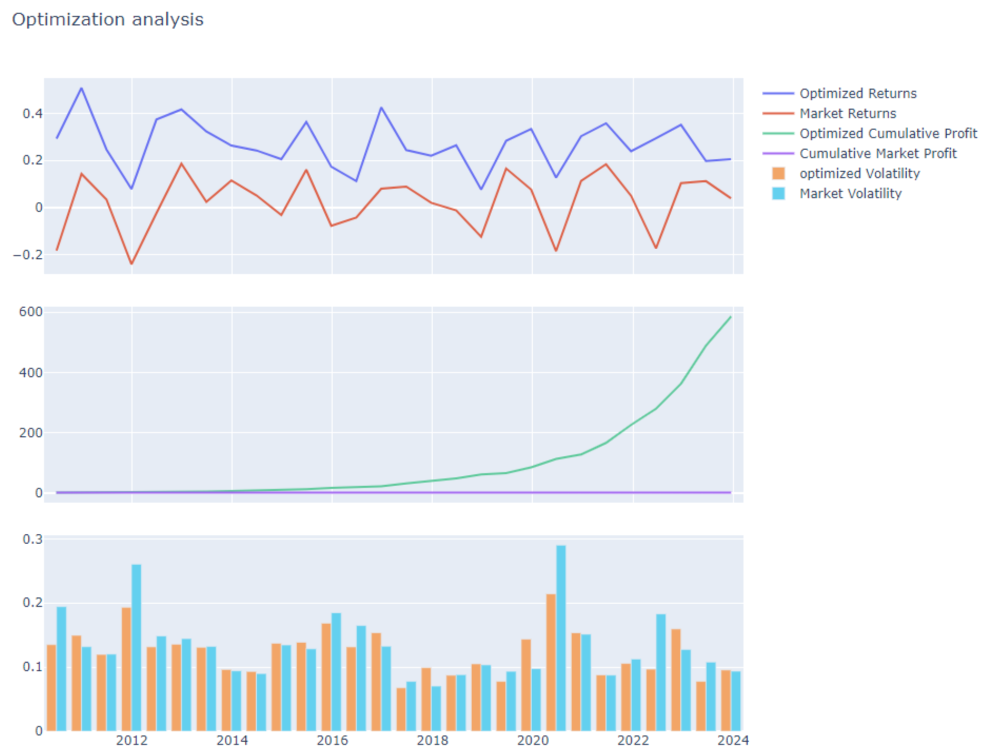


Figure 2 Returns, Cumulated Returns & Volatility of the Optimized Results and the Market

Figure 2 shows that the optimized results outperform the market in each category. The returns are always positive, even when the market performed poorly in 2019 and 2021 (due to the COVID crisis). The most impressive graphic compares the CAC 40 and the optimization cumulative result. The green function shows an exponential evolution and attains 600, which is the equivalent of earning x600 over the last 14 years.

These impressive returns are not attainable for real-life investors. The optimization program takes into consideration the returns of the semester i to maximize the Sharpe Ratio of the semester i. This would imply that the investor could see in the future, or at least predict with extreme precision the evolution of the stock market, which is impossible to do given the Brownian behavior of stock prices.

C. *Elaboration of a Buy/Sell Strategy*

While it is impossible to predict the most profitable stocks over some time T, it is possible to follow the evolution of these optimized stocks over $T + \Delta T$. The main hypothesis is that the best-performing stocks over T (in the context of the maximization of the Sharpe Ratio), may still provide positive results over the next period $T + \Delta T$. This is the reason why large capitalization companies are favored for this study, as the volatility of its stock prices are generally low, it minimizes the potential risk associated with holding them over the extended period of $T + \Delta T$.

The Buy/Sell Strategy for the rest of the simulations (if not contraindicated) will be the following:

- **Run** the simulation over a given period $\Delta T_{train} = [T_{train,start}, T_{train,end}]$
- **Buy** the indicated stocks with the associated weights at $T_{Train,end}$
- **Keep** the stock over the given period $\Delta T_{test} = [T_{test,start}, T_{test,end}]$

- **Sell** the entirety of the portfolio at $T_{test,end}$
- **Analize** the returns
- **Initialize** the new parameters for the next period

$$T^*_{train,start} = T_{train,end}$$
$$T^*_{train,end} = T^*_{train,start} + \Delta T_{train}$$
$$T^*_{test,start} = T_{test,end}$$
$$T^*_{test,end} = T^*_{test,start} + \Delta T_{test}$$

- **Repeat** the process for the next period

Let us note that it is logical that ΔT_{train} and ΔT_{test} succeed each other, therefore we can write $T_{train,end} = T_{test,start}$. Furthermore, the only parameters that have to be computed at the beginning of the simulation are : the starting date ($T_{train,0}$), the training period (ΔT_{train}), the testing/keeping period (ΔT_{test}) and the number of iterations n.

Note: This is not a traditional Machine Learning algorithm (excepted the minimization algorithm that can be considered as a form of Machine Learning), mostly because stock prices evolve with time. The reader must understand that in this simulation, there are no Train and Test sets but solely a minimization algorithm (that is used for decision making) as well as a “tracking” algorithm that calculates the true return for the next period.

IV. SIMULATION RESULTS AND ANALYSIS

A. First simulation and results

After setting the time parameters as the following:

$$T_{train,0} = 2010/01/01$$
$$\Delta T_{train} = 180\ days\ (1\ \text{semester})$$
$$\Delta T_{test} = 180\ days$$
$$n = 26\ (13\ \text{years})$$

This imposes the simulation to end around 12/2023. The optimization algorithm runs currently with additional bounds ($\forall k, w_k < 0.4$) in order to enlarge the number of selected stocks (minimum 3) for risk diminishing reasons. Here are the results:

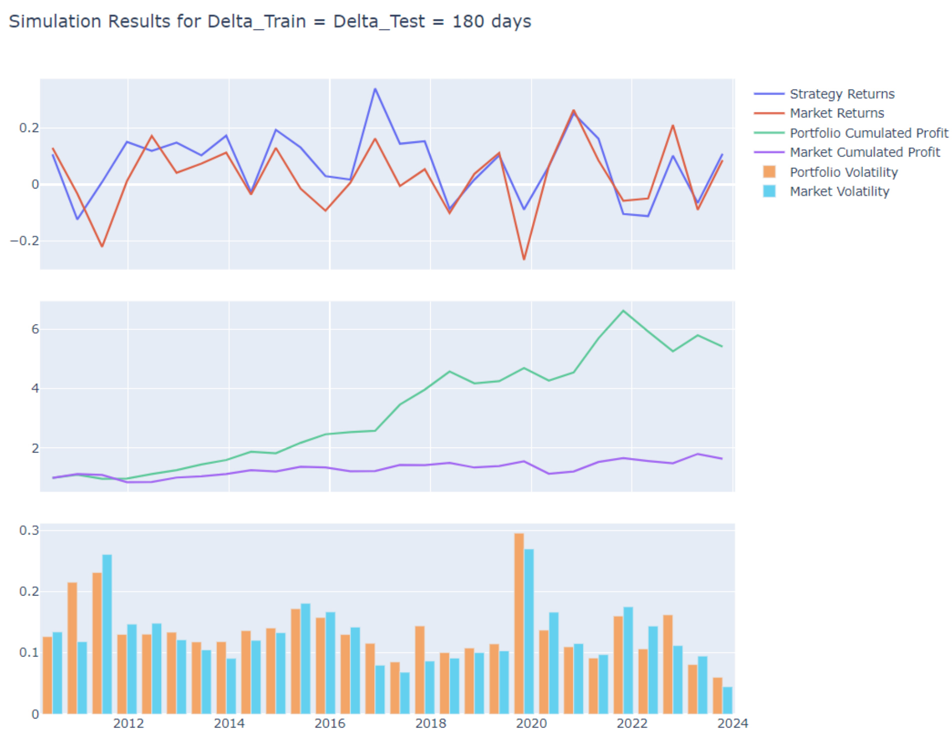


Figure 3 First Simulation Over 13 Years

In this simulation, the Strategy returns and the Portfolio Returns are resembling. However, the strategy outperforms the market by having an average return of 0.144/year against 0.056/year. The semester volatilities are similar in average: 0.131 and 0.129. The strategy performed better over time with a cumulated profit of 5.41 against 1.64 (non-adjusted to inflation). The cumulated profits graph is interesting to consider because it gives an idea on the consistency of the optimization method.

This is encouraging, because the simulation shows overall better results for the portfolio than for the CAC40, with a $\frac{Result}{Volatility}$ ratio clearly in favor of the optimization method. This also confirms the financial hypothesis that were made in the previous chapters.

B. Second Simulation and Potential Limitations

Positive results that outperforms the market on almost every semester raises the question of shortening ΔT_{test} while maintaining ΔT_{train} at its default value. This will raise the number of trades and presumably increase the returns.

This second simulation has the following parameters:

$$T_{train,0} = \mathit{2010/01/01}$$
$$\Delta T_{train} = 180 \; days \; \mathit{(1\ semester)}$$
$$\Delta T_{test} = 30 \; days$$
$$n = 316 \; \mathit{(13\ years)}$$

Every 30 days, the Buy/Sell strategy is resumed, and the results are based on the period $\Delta T_{train} = 180$ days. The results over the last 13 years are the following:

The execution of the simulation takes around 4 minutes to complete.

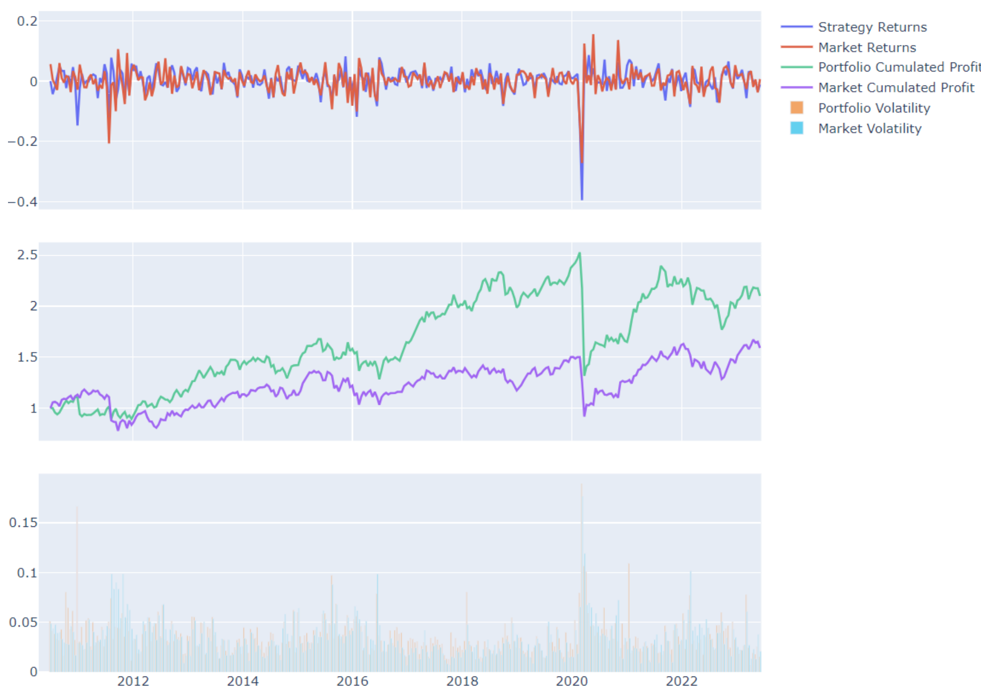


Figure 4 Simulation with T_train = 15 days

The graphics are more detailed and the run time takes longer because the time acquisition was divided by 12. The yearly returns for the portfolio is divided by 2 in comparison to the previous one (0.079). Unsurprisingly, he market returns remains the same (0.057). The cumulated returns over 13 years was divided nearly by 2 and dropped to 2.1.

C. Implementation of Selling Triggers

When ΔT_{test} is low (a couple days), the simulation and the market returns are strongly correlated. Furthermore, it is observed that when the index experiences a significant point drop (for instance during the COVID crisis[viii]), the simulation performs poorly (-40% at the start of 2020). This is an example of the limits of the Sharpe Ratio (refer to 2.B. *Drawbacks as a fund selection criteria*).

To mitigate this risk, we tend to employ "selling triggers," which automatically divest the entire portfolio when the CAC 40 returns fall below a specified threshold, known as the CAC_{limit}. The strategy resumes at the next iteration. In the case of this article, CAC_{limit} is initialized at -0.1.

The simulation is ran once again with the same parameters as 4.2 (and $CAC_{limit} = -0.1$).

Figure 5 Simulation and importance of filters

With this simulation, the loss related to the COVID crisis has been minimized and the strategy performs slightly better than the previous one. The yearly returns has improved up to 0.107. The volatility remains the same. However, this strategy is still less efficient than the first one.

V. PARAMETERS OPTIMIZATION

The objective of these next series of simulation is to determine the best parameters in order to maximize potential returns. In order to determine the best strategy, we will be focusing on the returns as well as the volatility ratio: $\beta = \frac{\sigma_{Portfolio}}{\sigma_{CAC40}}$, that should give an indication on the potential risk associated with said strategy compared to the market.

A. Optimizing ΔT_{train} & ΔT_{test}

After running the simulation for a couple of parameters (ΔT_{train}, ΔT_{test}), the results are documented in Table 1.

Experience	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Train (days)	180	180	180	180	720	720	360	360	180	180	90	90	45	45	90
Test (days)	180	15	15	720	720	360	360	180	180	90	90	45	45	90	180
Average returns	0.1446	0.078	0.107	0.073	0.093	0.117	0.127	0.133	0.132	0.0812	0.0605	0.064	0.086	0.0998	0.1122
Beta	1.016	1.059	1.059	0.963	1.087	1.059	1.053	1.031	1.045	1.049	1.011	1.002	1.031	1.049	1.066
Sell Triggers	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 1 Optimizing Simulation Parameters

To put into comparison, Table 2 indicates the average yearly returns of the CAC 40 along with its volatility.

CAC 40	
Average Return	0.065
Volatility	0.202

Table 2 Mean Yearly Log Returns Over Last 13 years & Volatility

In order to better understand Table 1, we can plot its components in a 3-dimensional graph, where the X and Y axis are respectively ΔT_{train} and ΔT_{test} and the Z axis represents the yearly log returns.

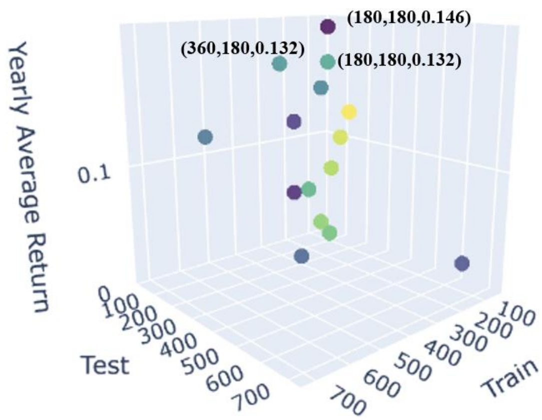


Figure 6 Yearly Average Return function of T_train & T_test

Surprisingly, the results demonstrate that the optimal parameters are the ones of the first simulation, followed closely by the same with the selling filter option activated. All in all, it appears that the optimal results are achieved with a particular value of ΔT_{test} = 180 days.

B. Explaining the value of $\Delta T_{test} = 180\ days$

The simulations with the parameter ΔT_{test} = 180 days are the one giving the optimal results. This particular value can be explained by a potential seasonality in the CAC 40 Market. In order to visualize this behavior, the following graph represents the average monthly returns divided by the volatility.

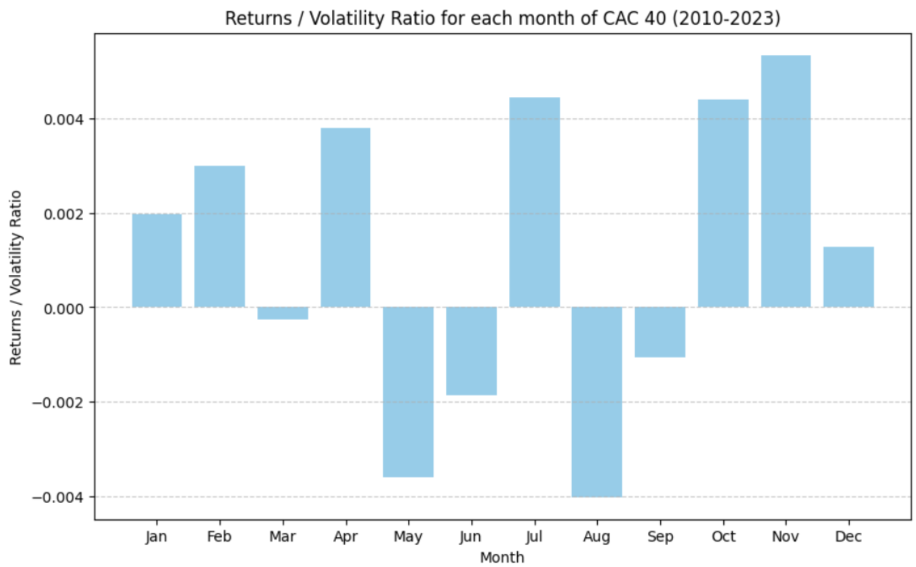


Figure 7 Returns/Volatility Ratio for each month of CAC 40 (2010 - 2023)

As depicted in Figure 7, the CAC 40 exhibits strong performance in October and November but tends to underperform in May and June. ΔT_{test} = 180 days suggests that the stocks are bought at the beginning of January and July and sold during the end of June and December. If no clear conclusion can be made by analyzing the first semester, the second semester shows very positive uptrends at the end of the year. In other words, the algorithm buys stock at a low price after 2 poor months of May and June and sells them at the end of the year when the prices came back up. This could explain such results, although it is likely not the sole cause.

VI. TESTING THE SIMULATION WORLDWIDE

This simulation is giving positive results when used in the French CAC 40, however it is important for a strategy to be able to perform in other markets, as possessing financial products worldwide can reduce the risk.

A. Backtesting on the S&P 500

The simulation 1a) will be testing the efficiency of this strategy in the American Market. The parameters used to run the simulation are identical to the first simulation ($\Delta T_{train} = \Delta T_{test} = 180\ days$), even if they might not be optimal.

The runtime of this simulation take significantly more time than with the CAC 40 companies due to the fact that there are 12.5x more companies in the S&P 500. In order to greatly reduce the run time of this algorithm this study will focus on the S&P 100 and suggest that the outcome will be somewhat equivalent. The updated runtime for this simulation takes around 3 minutes.

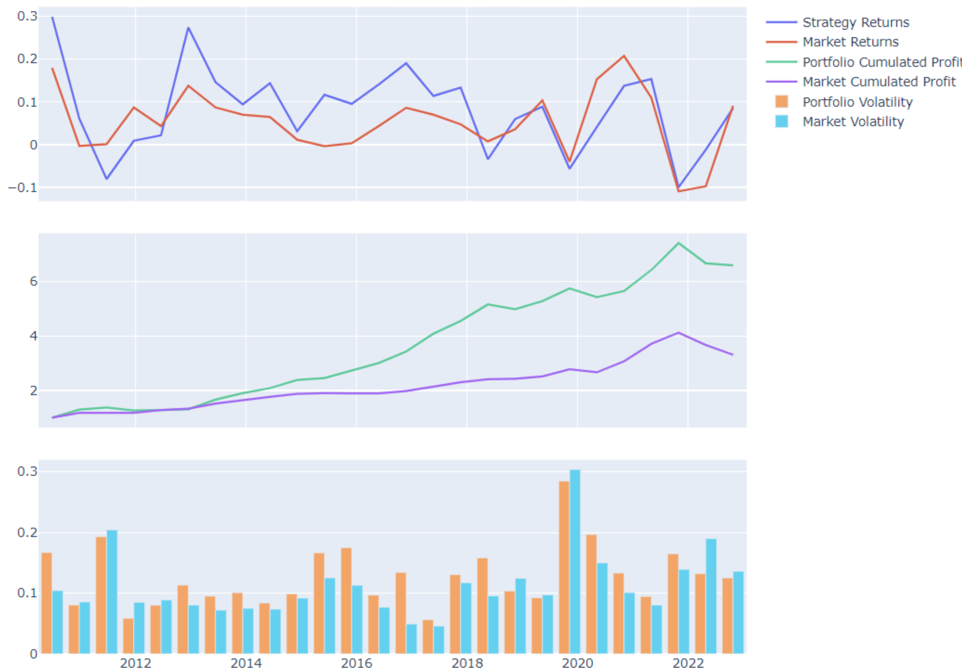


Figure 8 Simulation results for the S&P 100

The portfolio returns for this simulation are superior than any of the previous ones. The average yearly log returns is 0.166 for the portfolio whereas the market registers a yearly return of 0.106. The semester volatility is 0.128 against 0.112, but the returns over risk ratio is clearly in favor of the strategy. Finally, the cumulated profits over 13 years are also in favor of the strategy (x6.6 against x3.31), which indicates that the strategy is consistent over the years.

B. Verifying for any seasonality in the S&P 500

With the same method as V.B., the objective will be to confirm or deny the impact of seasonality on the portfolio optimization algorithm.

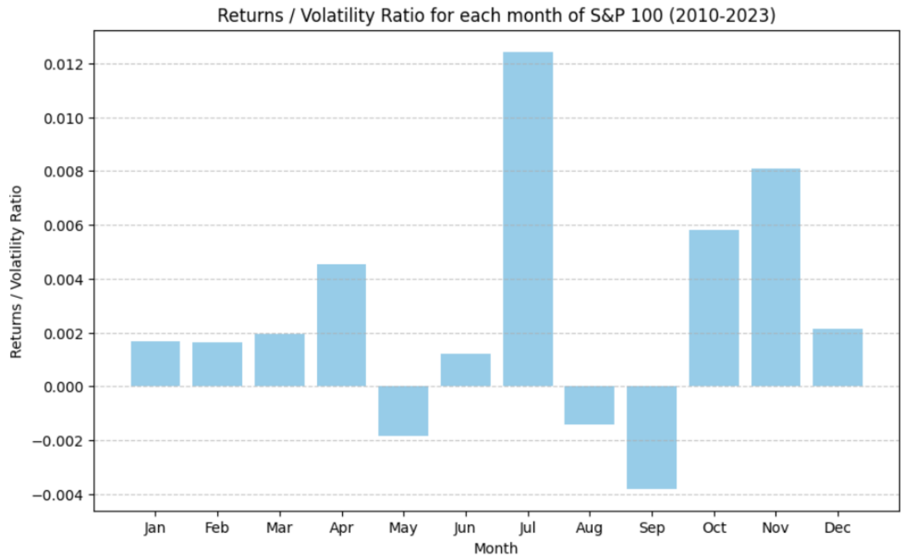


Figure 9 Returns Over Volatility Ratio for the S&P100

The S&P100's $\frac{Return}{Volatility}$ ratio for each month is somewhat similar to the CAC 40's, with a strong month of July as well as October and November. Therefore, it might explain why the parameters retained for the CAC 40 optimization algorithm works as well with the S&P 100.

C. Backtesting on the Chinese SSE50

The Shanghai Stock Exchange 50 Index is interesting to study as it has a high volatility compared to the other indexes upon which the model was backtested. Therefore, the returns should be lower because the Sharpe Ratio is less reliable under these circumstances. Adding to that, the SSE50 has lost near half of its value since 2021, showcasing the fact that the Chinese economy still experiences difficulties recovering from the Covid Crisis, this will make it harder to register positive returns. In this configuration, selling filters appears to be mandatory.

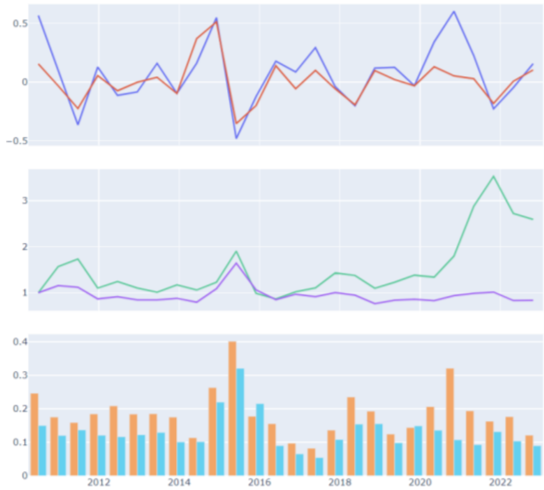


Figure 10 Simulation Results Without Selling Filters

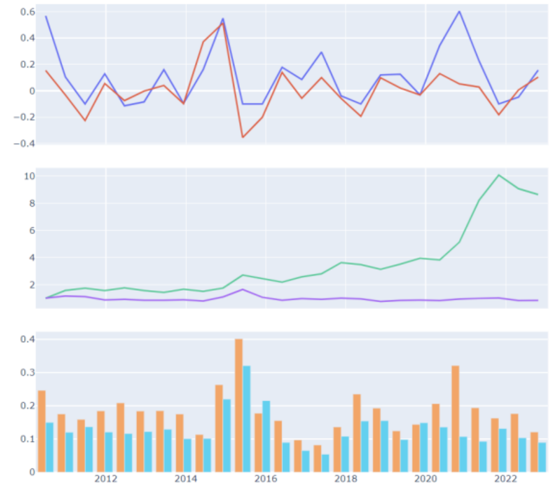


Figure 11 Simulation Results With Selling Filters

Contrary to the expected results, the simulation clearly outperformed the market. The selling filters limited the loss and the portfolio even reached a resounding 10x in cumulated returns at the end of 2021. This is due to the great results in 2020-2021(+0.25 and +0.6) that we will be trying to explain below.

```
            Optimization results 2020-01-01 00:00:00
        Selected Assets:
Poly Developments and Holdings Group Co., Ltd.: 0.2256
Jiangsu Hengrui Pharmaceuticals Co.,Ltd.: 0.0100
SHANXI XINGHUACUN FEN WINE FACTORY CO.,LTD: 0.0453
Zijin Mining Group Company Limited: 0.1687
WuXi AppTec Co., Ltd.: 0.1184
Will Semiconductor CO., Ltd. Shanghai: 0.2171
ZHEJIANG HUAYOU COBALT CO., LTD.: 0.1853
GigaDevice Semiconductor Inc.: 0.0296
```

Figure 12 Optimized Portfolio for the First Semester of 2020

```
            Optimization results 2020-06-30 00:00:00
        Selected Assets:
ZHANGZHOU PIENTZEHUANG PHARMACEUTICAL CO.,LTD.: 0.4000
SHANXI XINGHUACUN FEN WINE FACTORY CO.,LTD: 0.2505
China Tourism Group Duty Free Corporation Limited: 0.0560
Foshan Haitian Flavouring and Food Company Ltd.: 0.0781
Beijing Kingsoft Office Software, Inc.: 0.2155
```

Figure 13 Optimized Portfolio for the Second Semester of 2020

Figures 12 & 13 represent the selected companies associated with their weights for the first and second semesters of 2021. The next table represent the best evolutions during the holding period and should explain the impressive results.

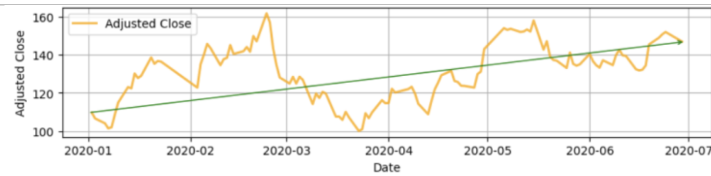
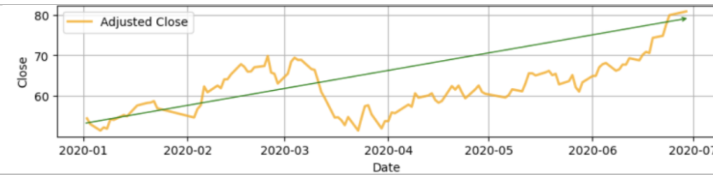
Company Name	% Change	Graphic Visualization
Will Semiconductor (22% portfolio)	+ 34%	
WuXi AppTec (11% portfolio)	+49%	

Table 3 Best evolutions during S1-2020

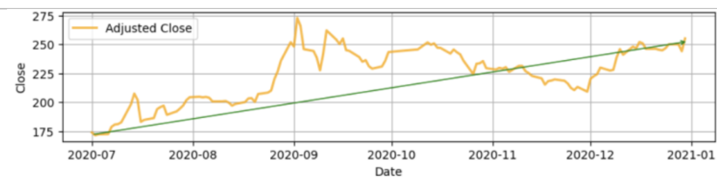
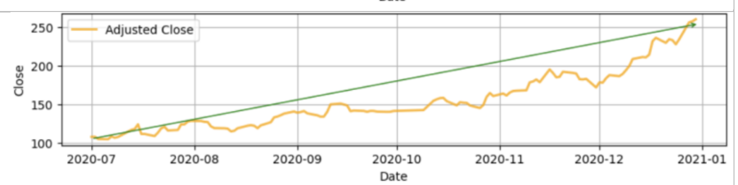
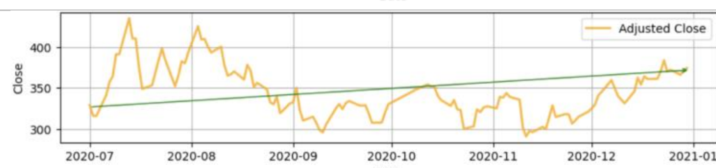
Company Name	% Change	Graphic Visualization
Zhangzhou Pientzehuang Pharmaceutical (40% portfolio)	+ 47%	
Shanxi Xinghuacun Fen Wine Factory (25%)	+140%	
Beijing Kingsoft Office Software Inc (22%)	+14%	

Table 4 Best Evolutions for S - II 2020

D. Historical values at risk

Determining the historical value at risk $HVaR$ is a way to evaluate the aggressiveness of a strategy by quantifying the potential loss in value of a portfolio over a specific time horizon at a given confidence level. The Value at risk is based on the returns of the strategy over the years. The following Table expresses the $HVaR$ for a time horizon of 180 days at a confidence level of 95% for the strategy on each market.

Table 5 Strategy Value at Risk

Market	Strategy $HVaR$	Time horizon	Confidence Level (%)
CAC 40	11 %	180	95
S&P 100	7.5 %	180	95
SSE 50	10%	180	95

The $HVaR$ is relatively important, demonstrating that this strategy is medium risk medium reward. The S&P appears to be, at the moment, the best market in which to invest in. The Value at Risk can be greatly mitigated by refining the model and comparing with other strategies. Furthermore, the $HVaR$ is calculated based on a 180 time horizon, which leaves the investor considerable amount of time to take adaptive actions in order to minimize the loss.

VII. DISCUSSIONS AND POSSIBLE IMPROVEMENTS

One of the limits of this model is that it primarily focuses on historical data to select appropriated weights. However, the market has been following an overall uptrend for many years, thus the efficiency of this method can be biased.

This part will focus on some ideas that can be implemented in order to further improve the algorithm or change the method of stock selection.

A. Equal Risk Contribution

The results show that stock picking methods with Sharpe Ratio optimization tends to have an important volatility overall. Equal Risk Contribution (ERC) is an allocation strategy that mitigates the risk by distributing equally among the assets of the portfolio. It is determined by calculating the risk of each asset and then assigning weights to each asset based on their risk contribution.

Let $\Sigma = (\sigma_1, \dots, \sigma_n)'$ be the estimated risk vector for all the n assets. The risk parity for each asset is:

$$RP_i = \frac{1}{\sigma_i} \ (i \in [1, n])$$

He associated weight is determined by:

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

ERC is often used by investors who want a more risk-balanced approach (however less potential earnings) to portfolio construction, especially in situations where Sharpe Ratio approaches might lead to concentrated risk in certain assets.

B. Toward Maximum Diversification

The method presented in this article is relatively similar to the *Maximum Diversification Method* explained by Yves CHOUEIFATY & Yves COIGNARD[ix], where the weights are optimized following the volatility rather than the returns.

The "Diversification Ratio" ($D(W)$) that is set to be optimized is:

$$D(W) = \frac{W'.\Sigma}{(P' cov_{\Sigma} P)^{\frac{1}{2}}}$$

Results show great returns for the Most-Diversified portfolio method, according to Yves Choueifaty and Yves Coignard, who ran their simulations during a period of 16 years.

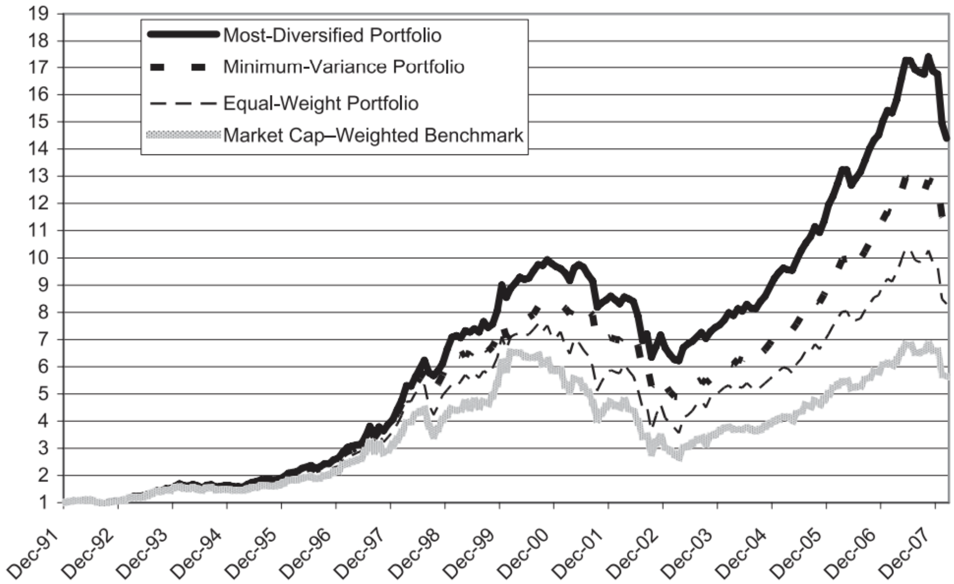


Figure 14 Comparison of Eurozone Equity Portfolios, 1992–2008

C. Constant Proportion Portfolio Insurance

Constant Proportion Portfolio Insurance (or CPPI) is designed to provide a floor of protection for an investment portfolio while still allowing for potential upside growth. This method adjusts the asset allocation strategy between risky assets and risk free assets based on the performance of the portfolio. Rather than implementing selling triggers as it is the case for the simulation presented in the article, using CPPI may help mitigate the loss during market downturns.

Let P_c be the value of the portfolio cushion and m the multiplier:

Then: Risky Asset Allocation $= P_c.m$

And: Risk-Free Asset Allocation = Portfolio Value - $P_c.m$

CPPI may prove useful as it provides a balance between risk and rewards and helps diminishing the overall value at risk of the portfolio.

CONCLUSION

It's crucial to bear in mind that the conclusions drawn are based on a computer simulation, and real-world implementation may vary. While this method is applicable on nearly every market in the world, it is advised to challenge the results with the reader’s own expertise of each stock. Ultimately, the guiding principle remains the importance of exercising common sense in investment decisions.

APPENDIX

While the entirety of the Python code will not be shared in this article for copyright reasons, this appendix will cover a part of it. It requires to input ΔT_{train} values and returns optimal stock for this period. This code does not offer any possibility to backtest the model, it is the reader's responsibility to do so. The author of this document reminds that the information provided is for educational and informational purposes only. The author is not a licensed financial advisor, and the content should not be construed as investment advice. Any actions taken based on the information provided are at the sole discretion and risk of the reader. The author shall not be held liable for any loss, financial or otherwise, arising from the use of or reliance on the information contained herein.

- It is advised to have an excel document with all the companies of a given market as well as their associated Tickers.
- The functions get_data() and optimize_portfolio() are not developed in this appendix.
- This code does not provide any possibility to evaluate the historical efficiency of the strategy, it is let to the reader to determine it.
- The libraries used for this simulation are listed below:

```
import numpy as np
import yfinance as yf
import pandas as pd
from scipy.optimize import minimize
from datetime import datetime, timedelta
from fredapi import Fred
```

```
path = r"C:\path_to_excel.xlsx"
api_key =   'api_key_for_risk_free_rate'

def get_data():
...
    return ...

def optimize_portfolio():
...
    return ...

def get_SNP_tickers():
    tickerOEX = pd.read_excel(path)
    ticker_list = list(tickerOEX['Ticker'])
    return ticker_list

def main():
# Input the Train Period
    start_date = input("Start Date: (YYYY-MM-DD): ")
    end_date = input("End Date: (YYYY-MM-DD): ")

    start_date = datetime.strptime(start_date, "%Y-%m-%d")
    end_date = datetime.strptime(end_date, "%Y-%m-%d")

# Get the intended tickers
    excel = pd.read_excel(path, index_col = 'Ticker')

    tickers = get_SNP_tickers()
    ad_close, log_returns = get_data(tickers, start_date, end_date)

# Get the latest Risk Free Rate
    fred = Fred(api_key='api_key')
    risk_free_rate = fred.get_series_latest_release('GS10').iloc[-1] / 100

# Optimization Process
    optimal_weights = optimize_portfolio(log_returns, risk_free_rate)

    selected_assets = []
    for ticker, weight in zip(log_returns.columns, optimal_weights):
        if weight > 1e-4:
            selected_assets.append((ticker, weight))

#Calculate the portfolio returns for the train period
    returns = 0
    for ticker, weight in selected_assets:
        returns += (np.sum(log_returns[ticker]))*weight

#Calculate the portfolio train test volatility
    train_volatility   =     np.sqrt(np.dot(optimal_weights.T,   np.dot(log_returns.cov()   *log_returns.shape[0],
optimal_weights)))

 #Print the results
     print(f'\n\n\t\tOptimization results {end_date}')
    print("\tSelected Assets:")
    for asset, weight in selected_assets:
        company_name = excel.loc[asset, 'Company Name']
        print(f"{company_name}: {weight:.4f}")

    print('\n\t\tResult')
    print('Result : ', round(returns, 3), '\tVolatility :', round(train_volatility, 3))
if __name__ == "__main__":
    main()
```

ACKNOWLEDGMENT

To be completed

REFERENCES

[i] Bachelier, L. (1900), «*Théorie de la spéculation*, Gauthier-Villars

[ii] Best, M.J. (2010), *"Portfolio Optimization"*, Chapman & Hall

[iii] Best, M.J. (2010), *"Portfolio Optimization"*, Chapman & Hall *: p59-78*

[iv] Sharpe, W. F. (1966). *"Mutual Fund Performance"*. Journal of Business*: p119-138*

[v] Scholz, Hendrik (2007). *"Refinements to the Sharpe ratio: Comparing alternatives for bear markets"*. Journal of Asset Management: *p347–357*

[vi] Scipy Guide : https://docs.scipy.org/doc/scipy/tutorial/index.html#user-guide

[vii] French Government definition of CAC 40: https://www.economie.gouv.fr/facileco/cac-40

[viii] Saira Naseer, Sidra Khalid, Summaira Parveen, Kashif Abbass, Huaming Song and Monica Violeta Achim. *"COVID-19 outbreak: Impact on global economy"*, Frontiers in Public Health

[ix] Yves Choueifaty, Yves Coignard, *Toward Maximum Diversification*, 2008