Response for Project Week 4

Course Code: Fintech545

Course Title: Quantitative Risk Management

Student Name: Wanglin (Steve) Cai

Student Net ID: WC191

Problem 1

Calculate and compare the expected value and standard deviation of price at time t (P_t), given each of the 3 types of price returns, assuming $r_t \sim N(0, \sigma^2)$. Simulate each return equation using $r_t \sim N(0, \sigma^2)$ and show the mean and standard deviation match your expectations.

1.1 Calculate and Compare the Expected Value and Standard Deviation of Price at Time t

In this question, at first it asks us to calculate and compare the expected value and standard deviation of price at time t.

For the **Classical Brownian Motion**, we have:

$$P_t = P_{t-1} + r_t$$

Therefore, we can use this formula to derive the expected value of price at time t:

$$E[P_t] = E[P_{t-1} + r_t]$$
$$E[P_t] = E[P_{t-1}] + E[r_t]$$
$$E[P_t] = P_{t-1}$$

Or by keep substituting:

$$E[P_t] = P_0$$

Also, we can use this formula to derive the standard deviation **given Pt-1**:

$$Std[P_t] = Std[P_{t-1} + r_t]$$
$$Std[P_t] = Std[P_{t-1}] + Std[r_t]$$
$$Std[P_t] = \sigma$$

Or we can calculate the standard deviation when we do not know Pt-1:

$$Std[P_t] = \sqrt{t} \times \sigma$$

For the **Arithmetic Return System**, we have:

$$P_t = P_{t-1}(1 + r_t)$$

Therefore, we can use this formula to derive the expected value of price at time t:

$$E[P_t] = E[P_{t-1} + P_{t-1}r_t]$$

$$E[P_t] = E[P_{t-1}] + E[P_{t-1}r_t]$$

$$E[P_t] = P_{t-1}$$

Or by keep substituting if we do not know the value of Pt-1:

$$E[P_t] = P_0$$

Also, we can use this formula to derive the standard deviation:

$$Std[P_t] = Std[P_{t-1} + P_{t-1}r_t]$$

$$Std[P_t] = Std[P_{t-1}r_t]$$

$$Std[P_t] = P_{t-1}\sigma$$

Or we can calculate the standard deviation when we do not know Pt-1:

$$Std[P_t] = \sqrt{t} \times P_0 \times \sigma$$

For the **Log Return or Geometric Brownian Motion**, we have:

$$P_t = P_{t-1}e^{rt}$$

Therefore, we can use this formula to derive the expected value of price at time t:

$$E[ln(P_t)] = E[ln(P_{t-1}e^{rt})]$$

$$E[ln(P_t)] = E[ln(P_{t-1})] + E[r_t]$$

$$E[ln(P_t)] = ln(P_{t-1})$$

Also, we can use this formula to derive the standard deviation:

$$Std[ln(P_t)] = Std[ln(P_{t-1}e^{rt})]$$

$$Std[ln(P_t)] = Std[ln(P_{t-1})] + Std[r_t]$$

$$Std[ln(P_t)] = \sigma$$

Comparing the three models, we can see that the standard deviation of the price process is proportional to the volatility in all three models, but the exact formula for the standard deviation differs. Specifically, the standard deviation is proportional to the square root of time in the classic Brownian motion and the arithmetic return system, but in the geometric Brownian motion, it depends on the exponential of the volatility. Besides, the standard deviation for the arithmetic return system is way larger than other two.

1.2 Simulate Each Return Equation and Show Mean and SD Match Expectations

We can simulate the given 3 types of price returns using Python code.

In this problem, we can simply assume the following:

$$P_{t-1} = 100, \ \ \sigma = 1$$

In this way, we can easily compute the result and show whether the mean and standard deviation match our expectations. For each of these methods, we can compute their expected value of Pt and the standard deviation of Pt.

For the Classical Brownian Motion, we have these result theoretically:

$$E[P_t] = 100, \ \ Std[P_t] = 1$$

For the Arithmetic Return System, we have these result theoretically:

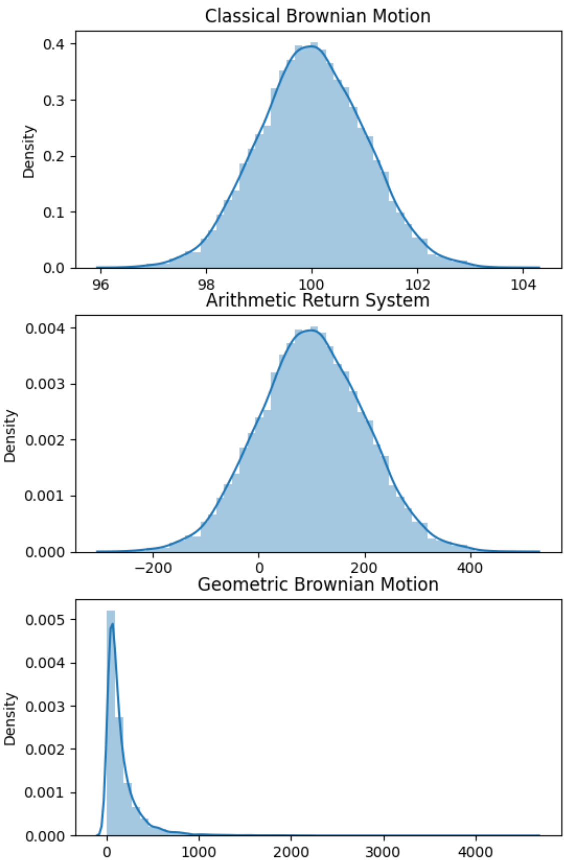
$$E[P_t] = 100, \ \ Std[P_t] = 100$$

For the Log Return or Geometric Brownian Motion, we have these result theoretically:

$$E[ln(P_t)] = 4.605, \ \ Std[ln(P_t)] = 1$$

Using Python, we can do the simulations. After 10000 times simulations, we can get the results of each method.

Firstly, the histogram of the price generated by each system is shown below:



For the Classical Brownian Motion, we get these results:

$$E[P_t] = 100.06, \quad Std[P_t] = 1.003$$

For the Arithmetic Return System, we get these results:

$$E[P_t] = 100.580, \quad Std[P_t] = 100.315$$

For the Log Return or Geometric Brownian Motion, we get these results:

$$E[ln(P_t)] = 4.611, \quad Std[ln(P_t)] = 1.003$$

By comparing the theoretical results and the simulated results, we can see that the results simulated generally match our expectations, the differences between them are neglectable.

Problem 2

Implement a function similar to the "return_calculate()" in this week's code. Allow the user to specify the method of return calculation.

Use DailyPrices.csv. Calculate the arithmetic returns for all prices.

Break the returns for META into 2 groups, a modeling group and a holdout sample. Make the holdout sample be the last 60 observations.

Remove the mean from the series so that the mean(META)=0

Calculate VaR
1. Using a normal distribution.
2. Using a normal distribution with an Exponentially Weighted variance ($\lambda = 0.94$)
3. Using a MLE fitted T distribution.
4. Using a fitted AR(1) model.
5. Using a Historic Simulation.

Compare the 5 values.

2.1 Implementation of function "return_calculate()"

From the code provided, I can implement the function using Python. The output of using arithmetic return is as follows:

	Date	SPY	AAPL	MSFT	AMZN	TSLA	GOOGL	GOOG	META	NVDA	...
1	2/15/2022 0:00	0.016127	0.023152	0.018542	0.008658	0.053291	0.007987	0.008319	0.015158	0.091812	...
2	2/16/2022 0:00	0.001121	-0.001389	-0.001167	0.010159	0.001041	0.008268	0.007784	-0.020181	0.000604	...
3	2/17/2022 0:00	-0.021361	-0.021269	-0.029282	-0.021809	-0.050943	-0.037746	-0.037669	-0.040778	-0.075591	...
4	2/18/2022 0:00	-0.006475	-0.009356	-0.009631	-0.013262	-0.022103	-0.016116	-0.013914	-0.007462	-0.035296	...
5	2/22/2022 0:00	-0.010732	-0.017812	-0.000729	-0.015753	-0.041366	-0.004521	-0.008163	-0.019790	-0.010659	...

5 rows × 101 columns

2.2 VaR Calculation

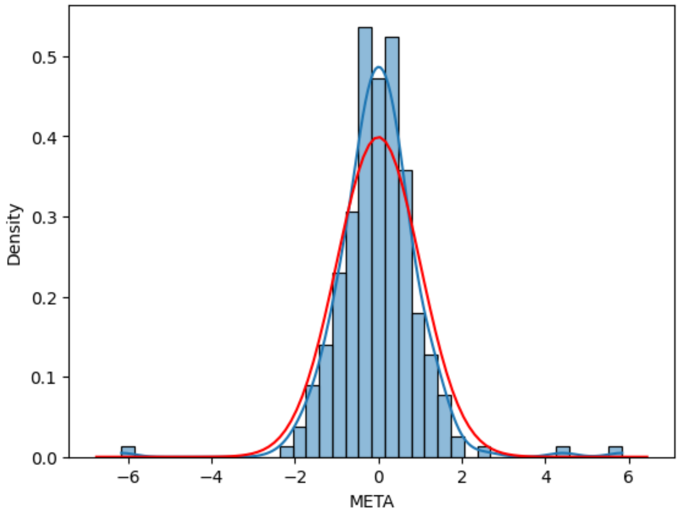
The next step, I calculated VaR using 5 methods mentioned in the problem 2. Firstly, we need to demean the return for 'META'.

```
# demean the return for META
Meta = all_return["META"] - all_return["META"].mean()
```

After demeaning the return, we can check that the return is zero. The reason why the result is not exactly zero is because the floating-point problem in Python.

```
1.6787646541711037e-18
```

Then, we calculate the standard deviation of this return array, under the assumption of normal distribution, we switched this array to standard normal and plot the histogram under the normal assumption.

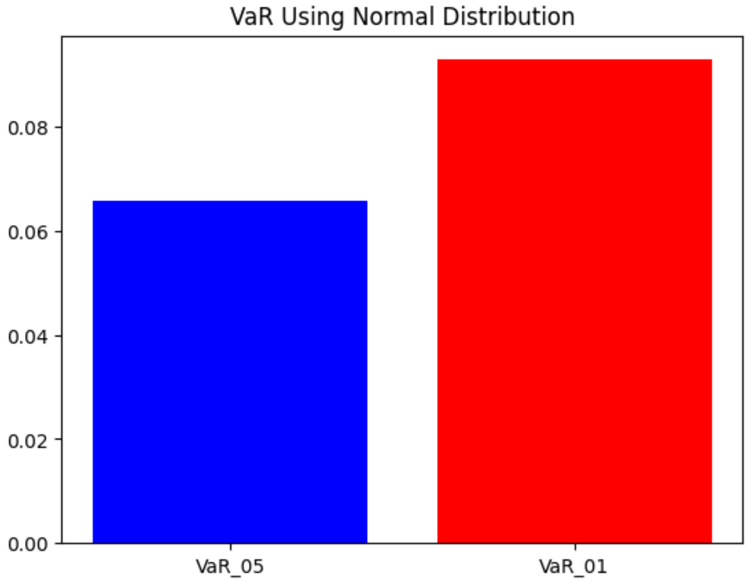


We found that the kurtosis of the distribution of the return is higher than standard normal distribution.

The first method is to calculate VaR **using a normal distribution**.

For 5% VaR, the value is 0.0656, which means that the minimum loss we can expect on a 5% bad day is 0.0656 (or 6.56%).

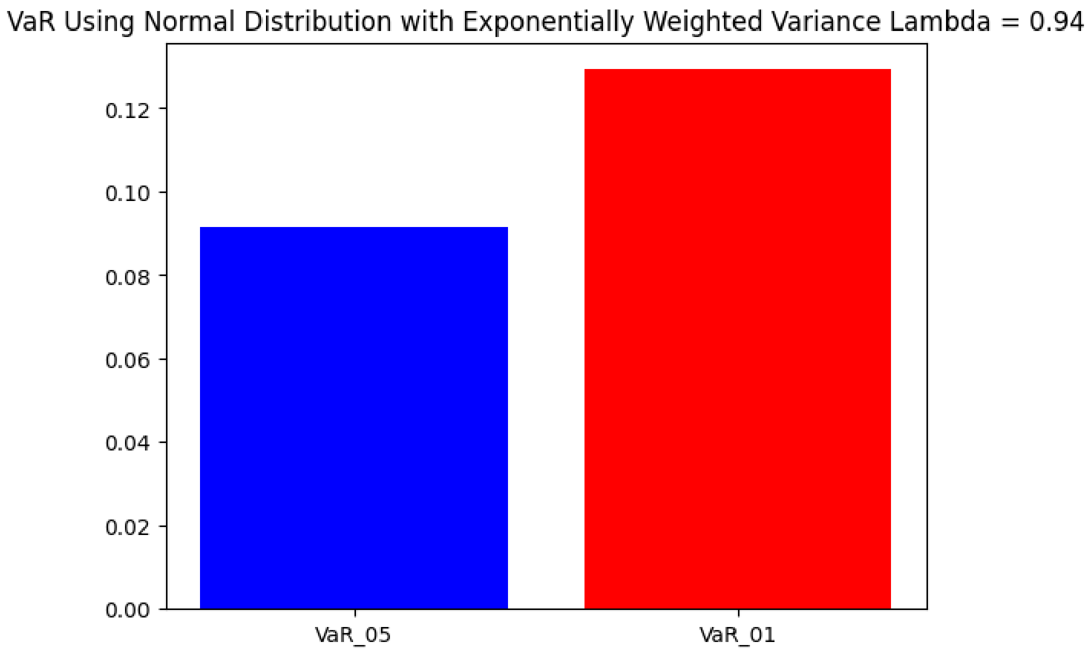
For 1% VaR, the value is 0.0928, which means that the minimum loss we can expect on a 1% bad day is 0.0928 (or 9.28%).



The second method is to calculate VaR **using a normal distribution with an Exponentially Weighted Variance with Lambda = 0.94**.

For 5% VaR, the value is 0.0913, which means that the minimum loss we can expect on a 5% bad day is 0.0913 (or 9.13%).

For 1% VaR, the value is 0.129, which means that the minimum loss we can expect on a 1% bad day is 0.129 (or 12.9%).



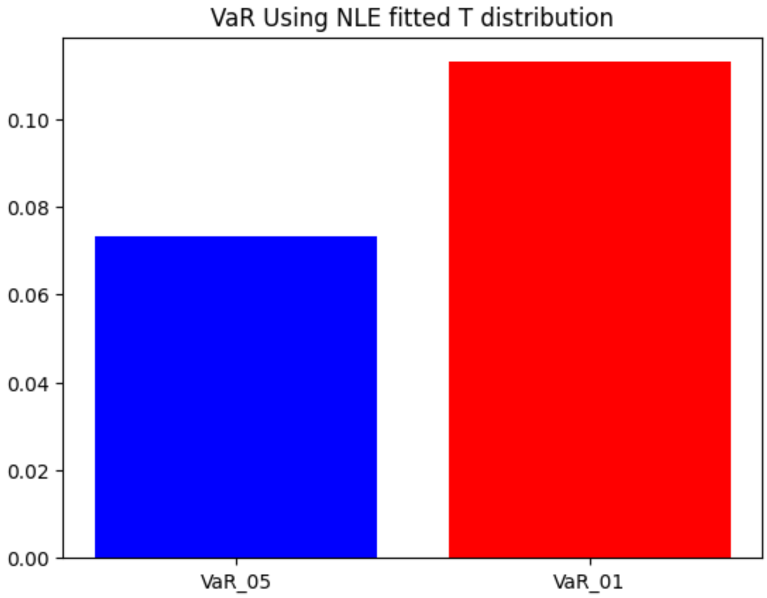
The result may reveal that the volatility is high relative to a year from now, thus the VaR here is higher than VaR using common variance.

The third method is to calculate VaR **using a MLE fitted T distribution**.

After fitting the T distribution, we found that the best df is 8.848.

For 5% VaR, the value is 0.0733, which means that the minimum loss we can expect on a 5% bad day is 0.0733 (or 7.33%).

For 1% VaR, the value is 0.113, which means that the minimum loss we can expect on a 1% bad day is 0.113 (or 11.3%).



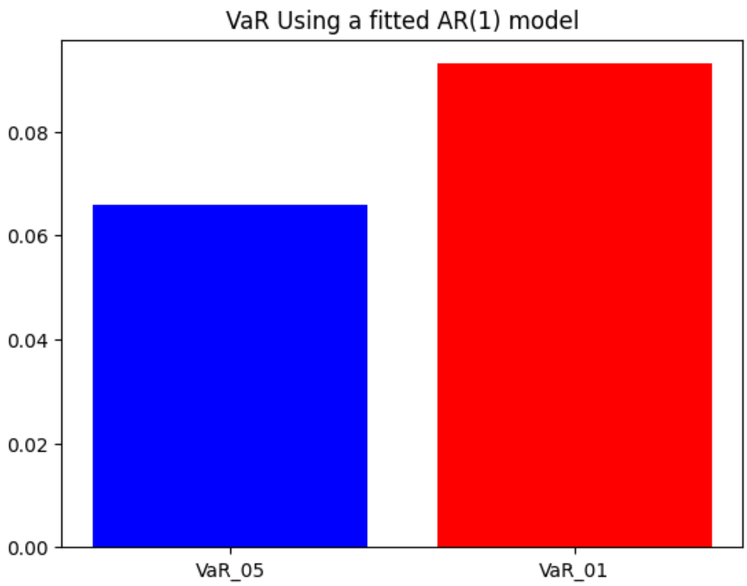
Since the t distribution has fat tiles at both end, it's easy to explain why the VaR is greater than the VaR using normal distribution.

The fourth method is to calculate VaR **using a fitted AR(1) model**.

After fitting an AR(1) model, since we assume that the white noise is a gaussian noise. It follows a normal distribution. Therefore, we can compute the VaR.

For 5% VaR, the value is 0.0659, which means that the minimum loss we can expect on a 5% bad day is 0.0659 (or 6.59%).

For 1% VaR, the value is 0.931, which means that the minimum loss we can expect on a 1% bad day is 0.931 (or 9.31%).



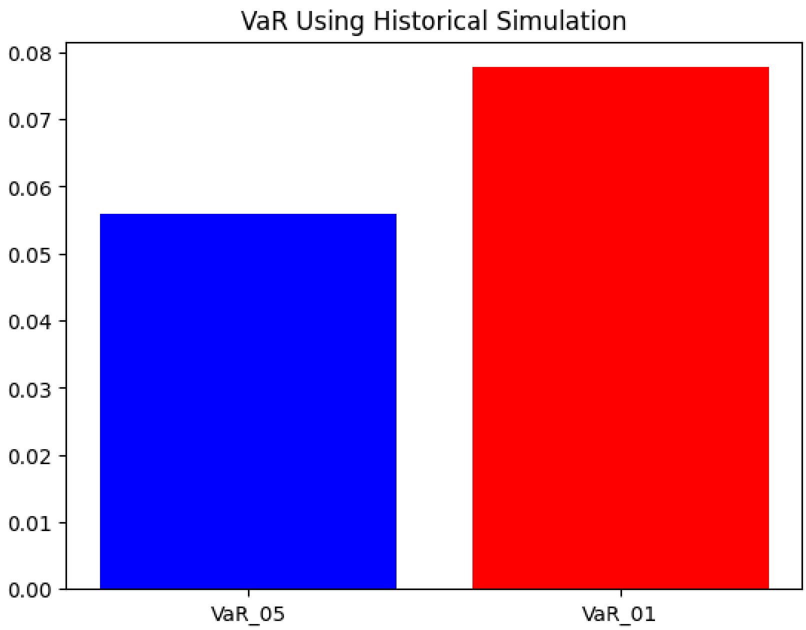
We can see that the VaR for fitted AR(1) model is roughly the same as the first model simply using normal distribution. One of the main reasons why the value of VaR is similar is because the parameters of the AR(1) model is small.

The fifth method is to calculate VaR **using a Historic Simulation**.

After simulating the N draws from history, we calculate VaR by these N draws. Since we assume that the weights for each day are equal, we just use uniform distribution to generate random number.

For 5% VaR, the value is 0.0559, which means that the minimum loss we can expect on a 5% bad day is 0.0559 (or 5.59%).

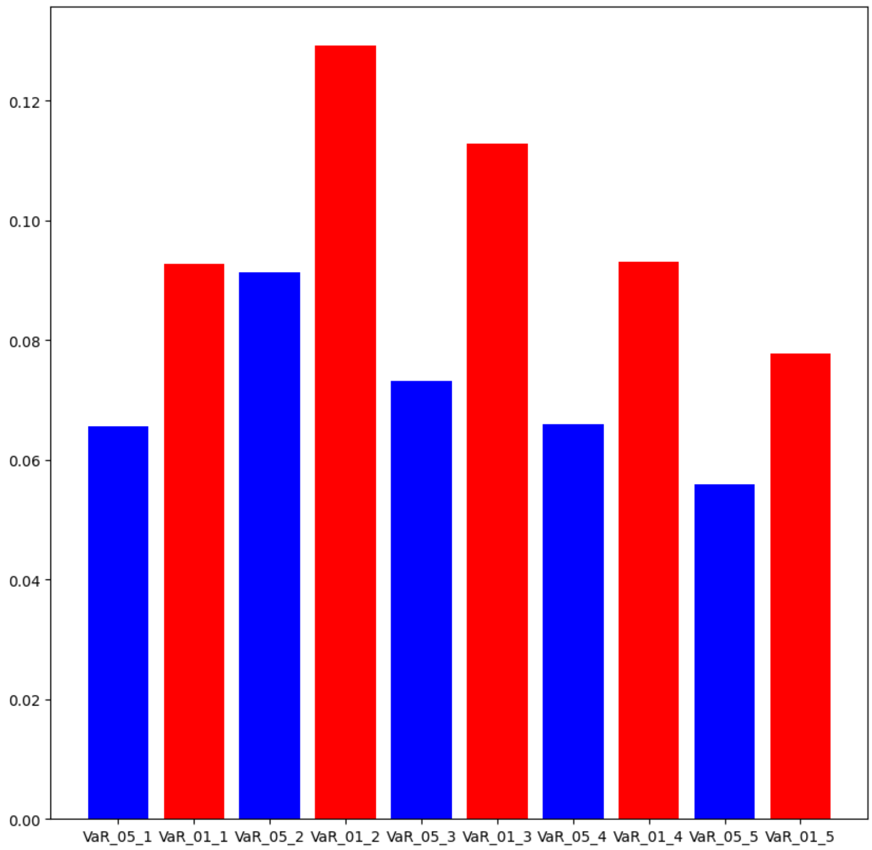
For 1% VaR, the value is 0.777, which means that the minimum loss we can expect on a 1% bad day is 0.777 (or 7.77%).



The results from this method probably indicate that the volatility of return is increasing from one year ago to now (or the stock market one year from now ran better than now, has more positive returns). If we get samples equally, since the bad returns are more from nowadays, and the good returns are more from long time ago, the result we get will not that up-to-date.

2.2 VaR Comparison

We can compare the VaR by displaying them on the same graph:



From this graph, we can see that the second method (normal with exponentially weighted variance) has the largest VaR, then the third method (MLE fitted T distribution). The reason why these two methods have larger VaR is because for the second method, it focuses more one the recent data since lambda is 0.94 in this problem. One reason behind the scenes is that the recent market is not as good as the market a year ago. And for the third method, it's because the T distribution is a fat-tail distribution, therefore, the VaR is larger naturally by its fat-tail property. For the last method, it can also be explained by the market reason. It's the opposite of the second method, whereas the second method consider more about the recent data. The last method consider the data equally throughout the year. That's why the VaR is much smaller than other methods.

Problem 3

Using Portfolio.csv and DailyPrices.csv. Assume the expected return on all stocks is 0.

This file contains the stock holdings of 3 portfolios. You own each of these portfolios. Using an exponentially weighted covariance with lambda = 0.94, calculate the VaR of each portfolio as well as your total VaR (VaR of the total holdings). Express VaR as a $.

Discuss your methods and your results.

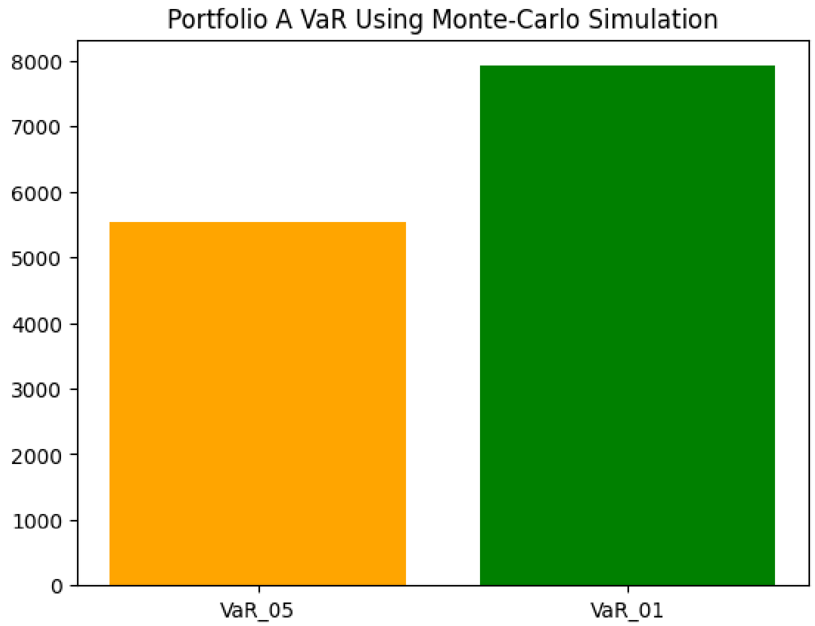
Choose a different model for returns and calculate VaR again. Why did you choose that model? How did the model change affect the results?

VaR Using An Exponentially Weighted Covariance with Lambda = 0.94

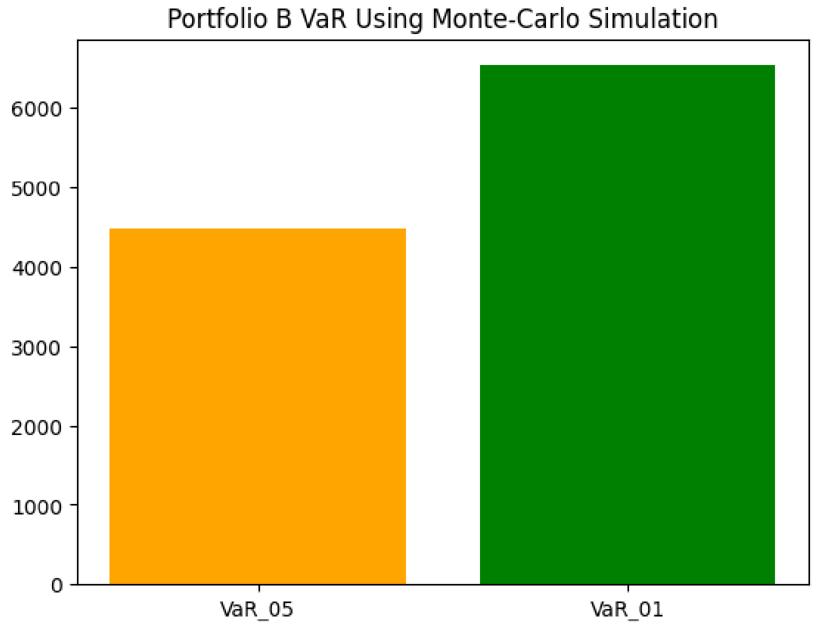
Monte-Carlo Simulations

First, we use Monte-Carlo simulations to calculate VaR for each portfolio with N = 10000. We use Monte-Carlo simulations because we are given the exponentially weighted covariance here in this problem. So we can use Monte-Carlo Simulations directly.

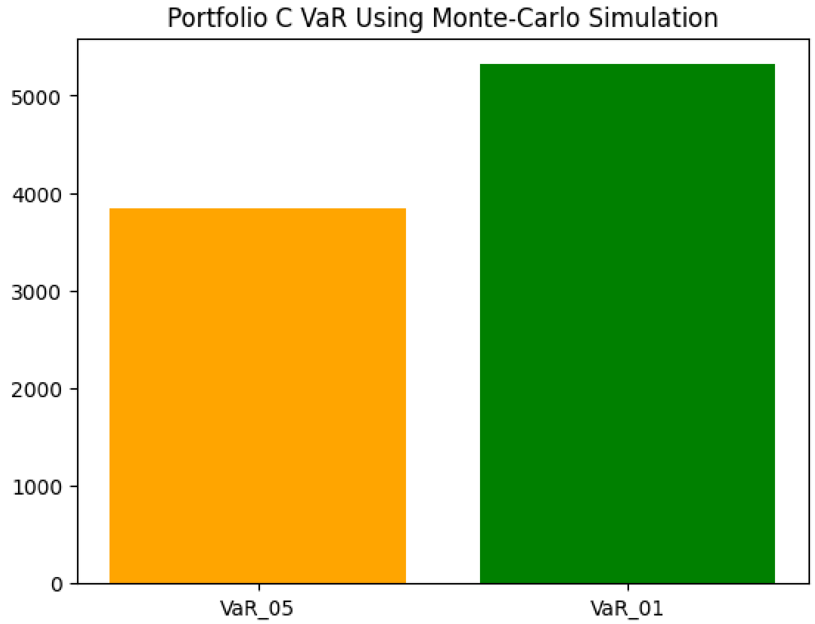
For Portfolio A, after simulation, the 5% VaR is $5539, which means that the minimum loss we can expect for Portfolio A on a 5% bad day is $5539. The 1% VaR, the value is $7923, which means that the minimum loss we can expect on a 1% bad day is $7923.



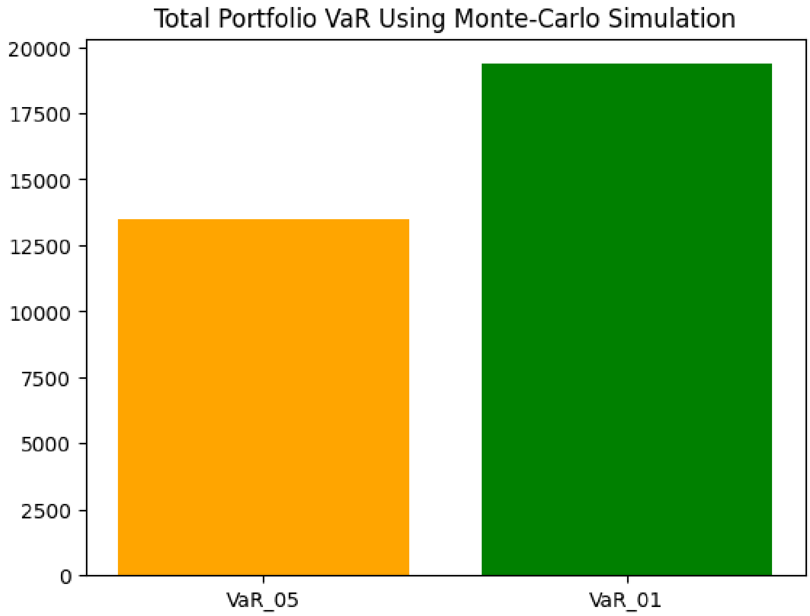
For Portfolio B, after simulation, the 5% VaR is $4486, which means that the minimum loss we can expect for Portfolio B on a 5% bad day is $4486. The 1% VaR, the value is $6536, which means that the minimum loss we can expect on a 1% bad day is $6536.



For Portfolio C, after simulation, the 5% VaR is $3846, which means that the minimum loss we can expect for Portfolio C on a 5% bad day is $3846. The 1% VaR, the value is $5318, which means that the minimum loss we can expect on a 1% bad day is $5318.



Then, we calculate the total portfolio's VaR. For Total Portfolio, after simulation, the 5% VaR is $13487, which means that the minimum loss we can expect for Total Portfolio on a 5% bad day is $13487. The 1% VaR, the value is $19354, which means that the minimum loss we can expect on a 1% bad day is $19354.



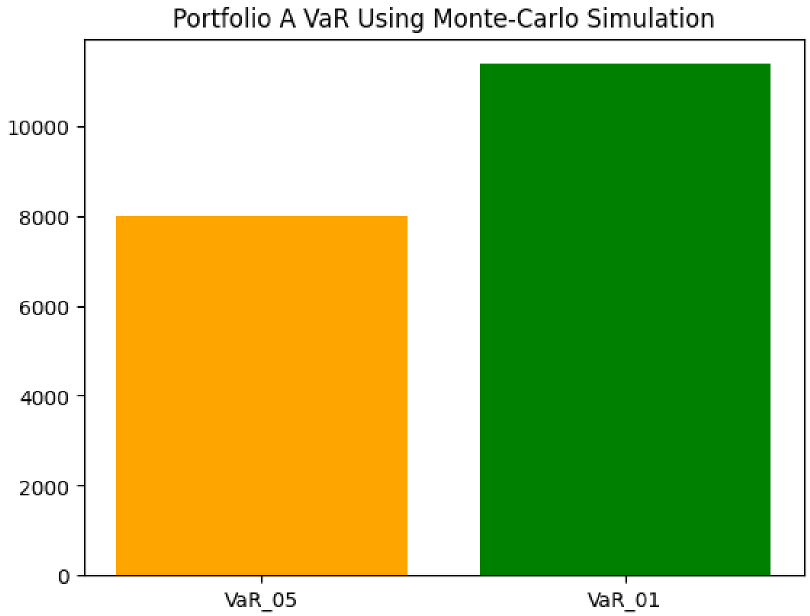
VaR Using Original Covariance

Monte-Carlo Simulations

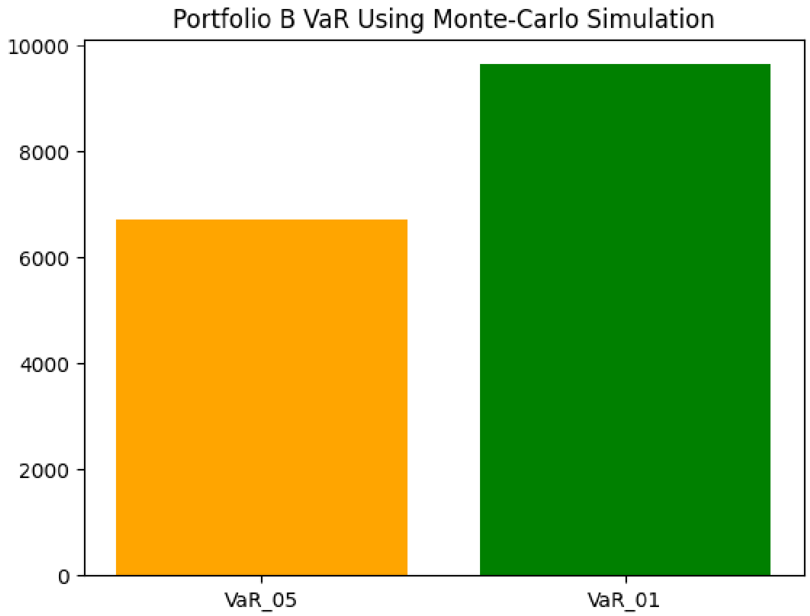
Here, we use a different model compared to the exponentially weighted covariance model. The model here is we simply use the covariance of the return. **The reason** we choose this model is that this method takes into account the older data equally rather than focuses more on the recent data. It's more like a comprehensive view of the past year risk/volatility.

We use Monte-Carlo simulations to calculate VaR for each portfolio with N = 10000.

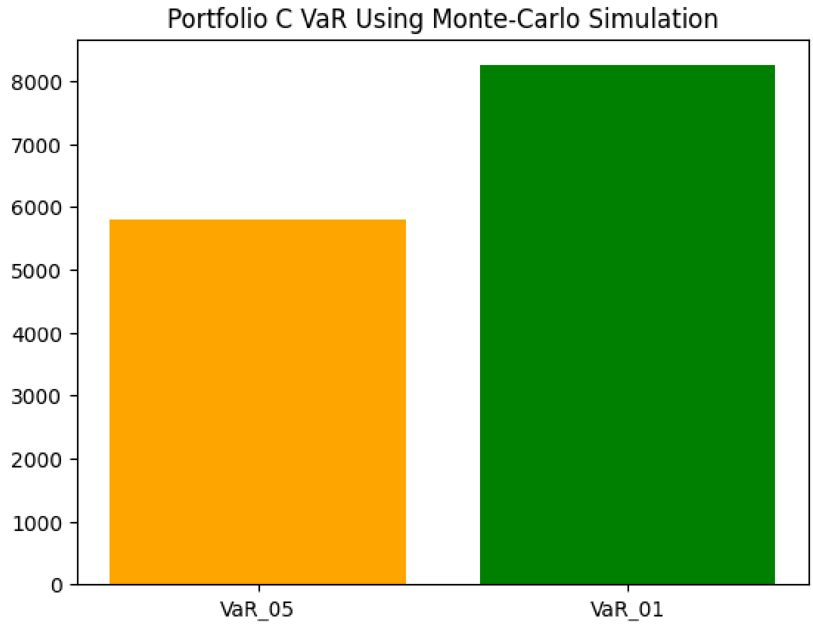
For Portfolio A, after simulation, the 5% VaR is \$ 7972, which means that the minimum loss we can expect for Portfolio A on a 5% bad day is \$ 7972. The 1% VaR, the value is \$ 11369, which means that the minimum loss we can expect on a 1% bad day is \$ 11369.



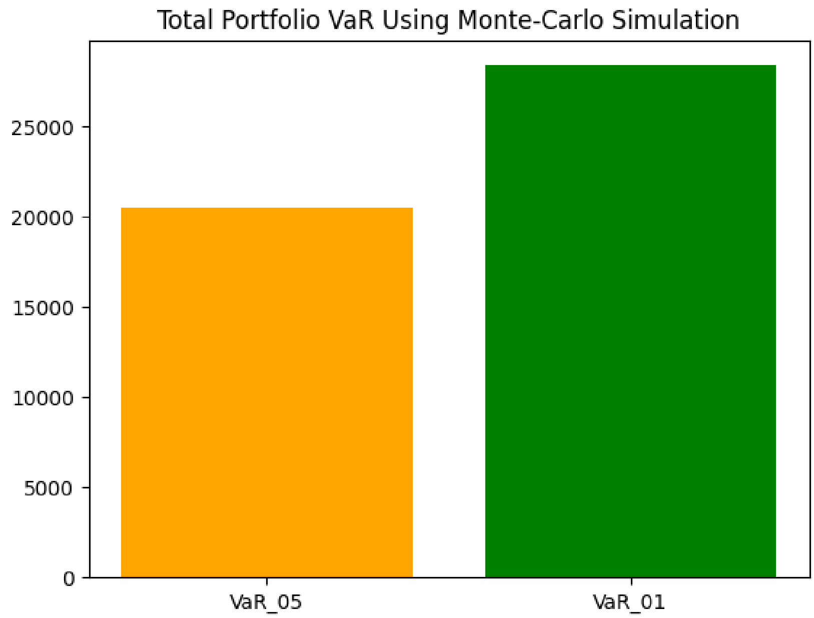
For Portfolio B, after simulation, the 5% VaR is $ 6716, which means that the minimum loss we can expect for Portfolio B on a 5% bad day is $ 6716. The 1% VaR, the value is $ 9641, which means that the minimum loss we can expect on a 1% bad day is $ 9641.



For Portfolio C, after simulation, the 5% VaR is $ 5797, which means that the minimum loss we can expect for Portfolio C on a 5% bad day is $ 5797. The 1% VaR, the value is $ 8248, which means that the minimum loss we can expect on a 1% bad day is $ 8248.



Then, we calculate the total portfolio's VaR. For Total Portfolio, after simulation, the 5% VaR is $ 20442, which means that the minimum loss we can expect for Total Portfolio on a 5% bad day is $ 20442. The 1% VaR, the value is $ 28327, which means that the minimum loss we can expect on a 1% bad day is $ 28327.

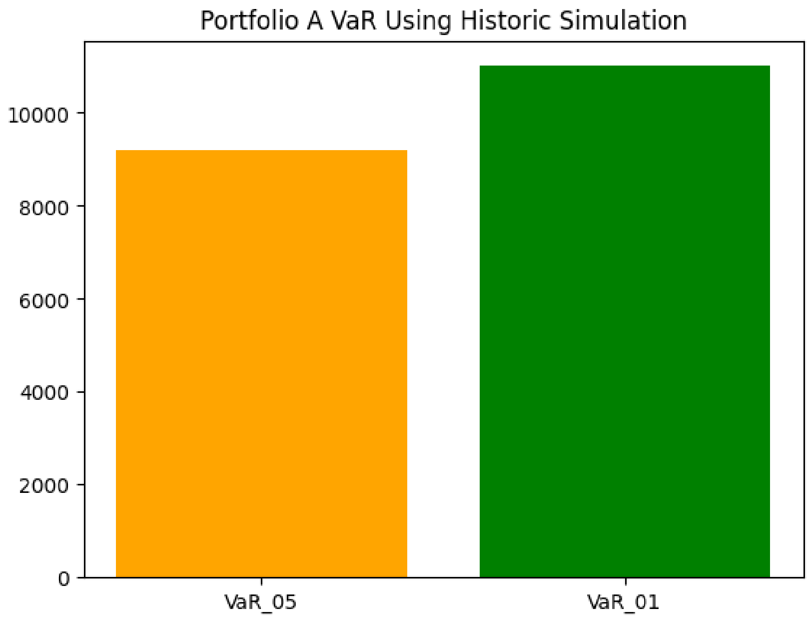


We can see that the results from this method are larger than the result using the exponentially weighted covariance. We know that the exponentially weighted covariance method focuses more on recent data. Based on this point of view, it is more likely that the selected portfolio's performance is bad recently compared to its performance a year ago. The original covariance method is more like a comprehensive view of the risk in a long run compared to the weighted method. We need to know that VaR is just a tool of quantifying risk. Most of the time we need to calculate and compare the VaRs using different methods to get a comprehensive knowledge of our portfolio's risk.

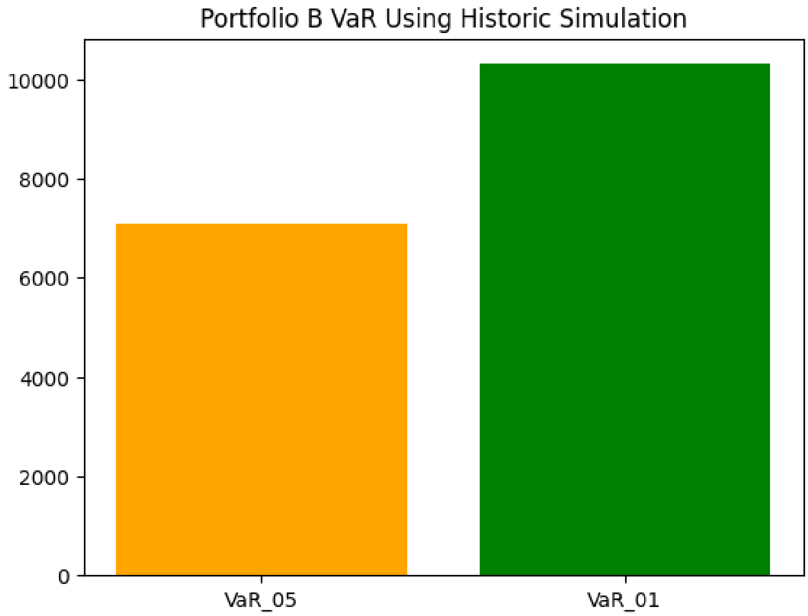
Other Methods of Simulation: Historic Simulation

In addition, there are also simulation methods other than Monte-Carlo, e.g., Historic simulations, to calculate VaR for each portfolio with N = 10000.

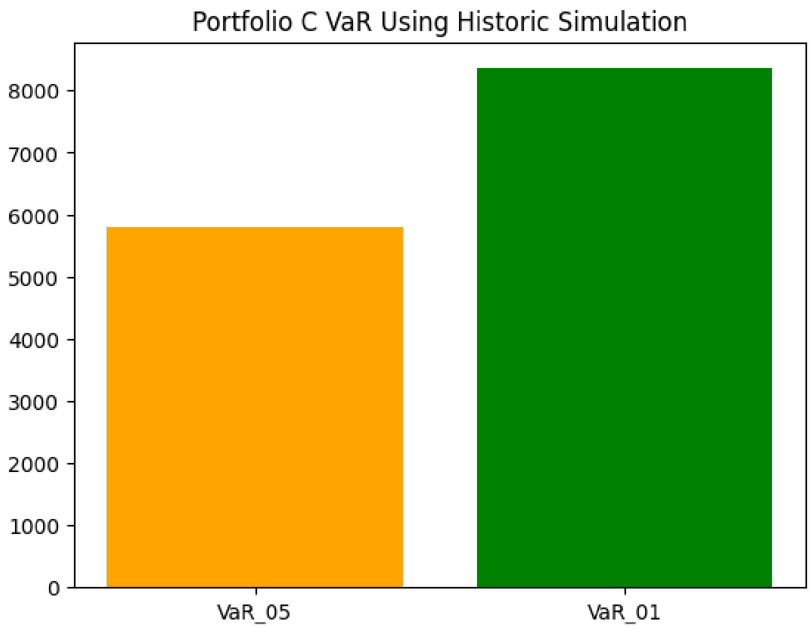
For Portfolio A, after simulation, the 5% VaR is $ 9070, which means that the minimum loss we can expect for Portfolio A on a 5% bad day is $ 9070. The 1% VaR, the value is $ 11005, which means that the minimum loss we can expect on a 1% bad day is $ 11005.



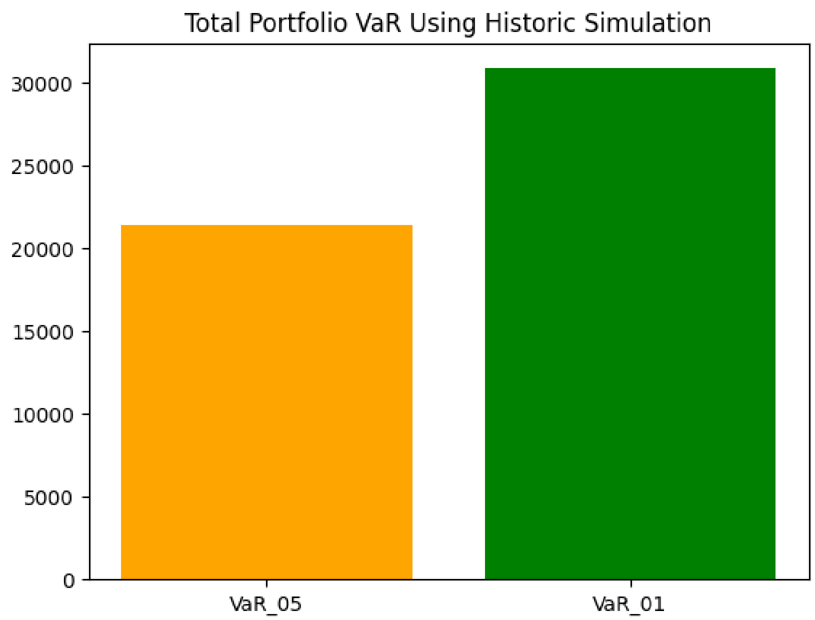
For Portfolio B, after simulation, the 5% VaR is $ 7078, which means that the minimum loss we can expect for Portfolio B on a 5% bad day is $ 7078. The 1% VaR, the value is $ 10302, which means that the minimum loss we can expect on a 1% bad day is $ 10302.



For Portfolio C, after simulation, the 5% VaR is \$ 5802, which means that the minimum loss we can expect for Portfolio C on a 5% bad day is \$ 5802. The 1% VaR, the value is \$ 8355, which means that the minimum loss we can expect on a 1% bad day is \$ 8355.



Then, we calculate the total portfolio's VaR. For Total Portfolio, after simulation, the 5% VaR is \$ 21408, which means that the minimum loss we can expect for Total Portfolio on a 5% bad day is \$ 21408. The 1% VaR, the value is \$ 30824, which means that the minimum loss we can expect on a 1% bad day is \$ 30824.



In the result, we can found that the VaR calculated by Historic Simulations is larger than the covariance method. However, it is similar to the covariance method. It is easy to explain this because we assign equal weight to each day in the Historic Simulations. Under this circumstances, it is similar to the original covariance method in some sense. Since they both treats equally to the whole data.

Since the historic simulations does not assume the distribution of the return, it provides another way for us to understand the risk of our portfolio. Because sometimes it's hard to assume the distribution of the return (we don't have expert knowledge of the distribution of the return). Therefore, we can use Historic Simulations to calculate VaR.