# Quantitative Risk Management Project 3

Qijun Yang
Duke University
Financial Technology at Duke University

*Version: 1.1*

*Date: Feb. 11, 2023*

## Problem 1

Use the stock returns in DailyReturn.csv for this problem. DailyReturn.csv contains returns for 100 large US stocks and as well as the ETF, SPY which tracks the S&P500.

Create a routine for calculating an exponentially weighted covariance matrix. If you have a package that calculates it for you, verify that it calculates the values you expect. This means you still have to implement it. Vary $\lambda$ ranging from 0 to 1. Use PCA and plot the cumulative variance explained by each eigenvalue for $\lambda \in (0, 1)$ each chosen.

What does this tell us about values of and the effect it has on the covariance matrix?

## Answer

**Model (Exponentially Weighted Volatility):**

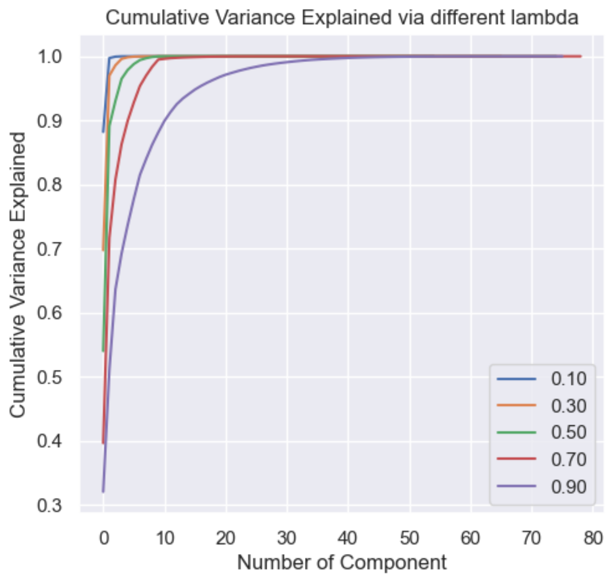
$$\sigma_t^2 = \lambda \sigma_{t-1}^2 + (1 - \lambda) r_{t-1}^2$$

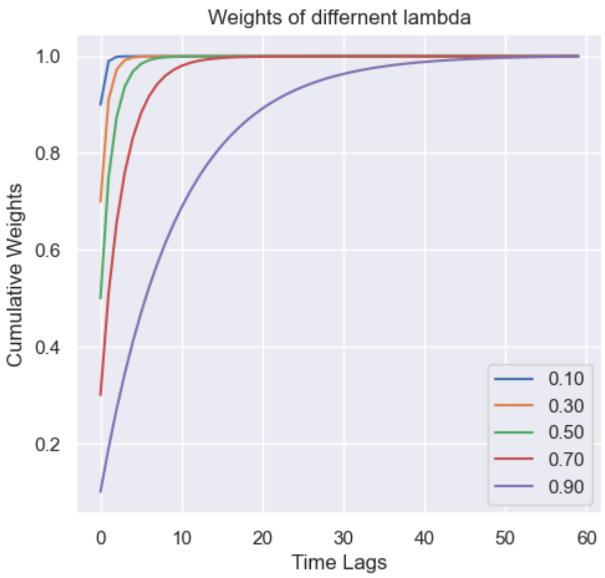
Given the data, we choose $\lambda$ to be 0.1,0.3,0.5,0.7,0.9 and compute the corresponding exponentially weighted covariance matrix of the stock returns of different firms and the weights of different time lags. Then we use PCA to decompose the covariance matrix, which gives us ordered eigenvalues and eigenvectors. Using the results above, we could plot the following images.

We can easily see that as $\lambda$ increases, the initial value of the cumulative variance explained also increases. With larger $\lambda$ the whole cumulative variance explained curve moves towards the point(0,1). If we look at the second plot, we can see why this happens. The larger $\lambda$, the more evenly the weight of the different time lags is distributed. The smaller $lambda$, the more weight is given to recent stock returns.

**Therefore, all of them make sense. A larger $\lambda$ means that the information is more likely to be evenly distributed. Smaller $\lambda$ means that recent data contains more information and the information is**

**concentrated in one place. When we do the PCA, we could use fewer principal components to represent the whole data if $\lambda$ is large, and vice versa.**





## Problem 2

Copy the chol_psd(), and near_psd() functions from the course repository - implement in your programming language of choice. These are core functions you will need throughout the remainder of the class

Implement Higham's 2002 nearest psd correlation function.

Generate a non-psd correlation matrix that is 500x500. You can use the code I used in class:

```
n = 500
sigma = fill(0.9, (n, n))
for i in 1:n
    sigma[i, i] = 1.0
end
sigma[1, 2] = 0.7357
sigma[2, 1] = 0.7357
```

Use near_psd() and Higham's method to fix the matrix. Confirm the matrix is now PSD.

Compare the results of both using the Frobenius Norm. Compare the run time between the two. How does the run time of each function compare as N increases?

Based on the above, discuss the pros and cons of each method and when you would use each. There is no wrong answer here, I want you to think through this and tell me what you think.

## Answer

After we convert the non-positive semidefinite matrix to a positive semidefinite matrix by Rebonato and Jackel's method, and Higham's method, we calculate all the eigenvalues of the generated positive semidefinite matrix and find that they are all positive under certain tolerance. Also, we use the previous Cholesky algorithm to test it and find that it's all Positive Semi-Definite Matrix.

Given a non-positive semidefinite matrix, we separately compute the nearly positive semidefinite matrix by gradually increasing the matrix size. We use 0,50,100,200 as our matrix size and find that as the size increases, the running time of both increases, and the Weighted Frobenius Norm of both also increases. However, Higham's method takes more time than Rebonato and Jackel's method, and Higham's method is more accurate. Higham's method consumes more computational power and gives better results.

Therefore, we could safely conclude:

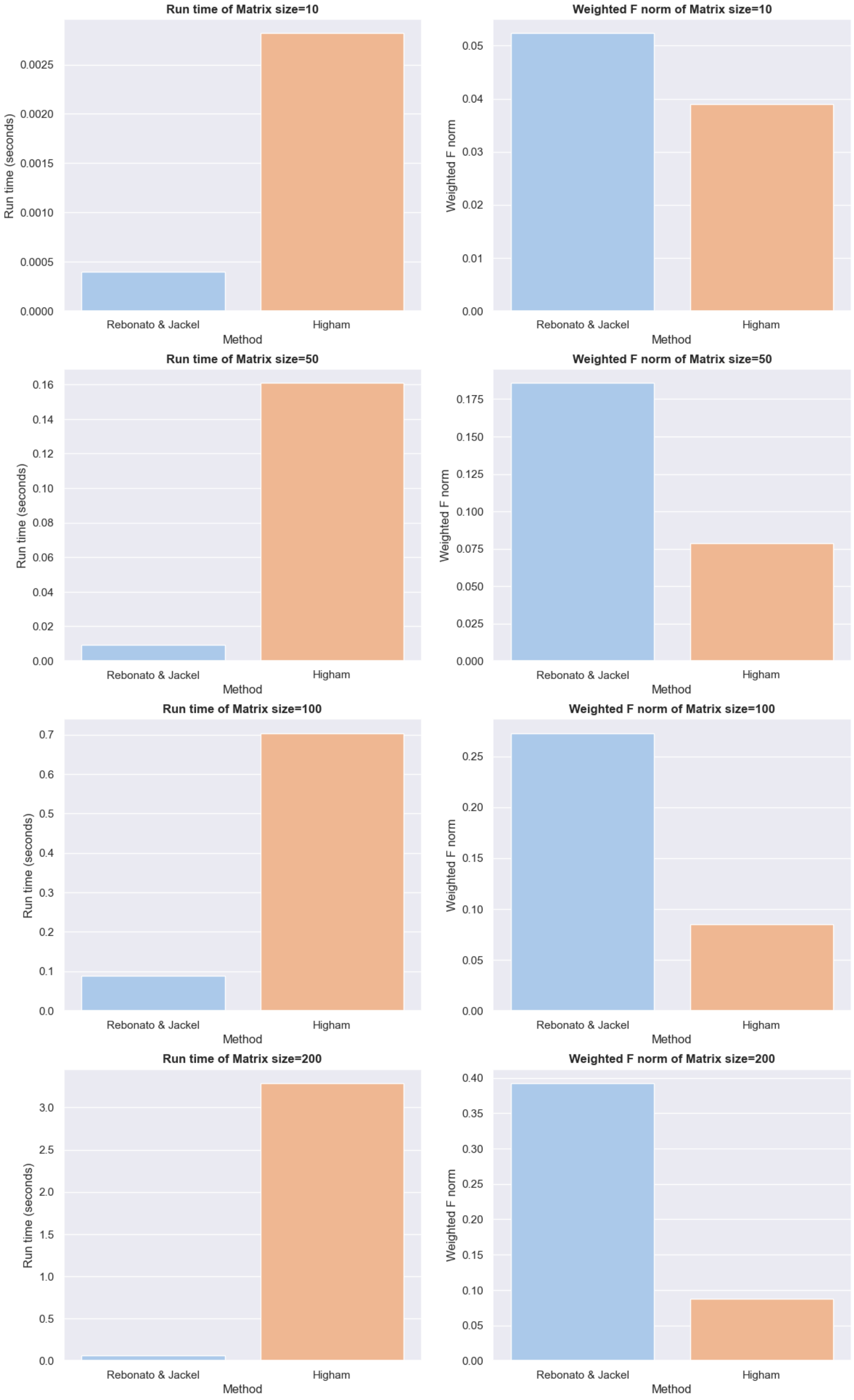
**For Rebonato and Jackel's Method:**

1. Prons: Fast and Easy to write
2. Cons: The precision is not the smallest

**For Higham's Method:**

1. Prons: High Precision
2. Cons: Consume more computation resources and comparatively slow

When the matrix size is small, both are fast, but Higham's method has higher accuracy. So I'll choose Higham's method when the matrix size is not very large. When the matrix size is much larger, the run time of Higham's method increases faster and the accuracy is maintained. There is a trade-off between time spent and precision obtained. Therefore, if the time is sufficient and we need high precision of the PSD matrix, I would like to choose Higham's method. Otherwise, I prefer Rebonato and Jackel's method.


Run time of Matrix size=10
Run time (seconds)
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
Rebonato & Jackel
Higham
Method
Weighted F norm of Matrix size=10
Weighted F norm
0.00
0.01
0.02
0.03
0.04
0.05
Rebonato & Jackel
Higham
Method
Run time of Matrix size=50
Run time (seconds)
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
Rebonato & Jackel
Higham
Method
Weighted F norm of Matrix size=50
Weighted F norm
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
Rebonato & Jackel
Higham
Method
Run time of Matrix size=100
Run time (seconds)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Rebonato & Jackel
Higham
Method
Weighted F norm of Matrix size=100
Weighted F norm
0.00
0.05
0.10
0.15
0.20
0.25
Rebonato & Jackel
Higham
Method
Run time of Matrix size=200
Run time (seconds)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Rebonato & Jackel
Higham
Method
Weighted F norm of Matrix size=200
Weighted F norm
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Rebonato & Jackel
Higham
Method

## Problem 3

Using DailyReturn.csv.

Implement a multivariate normal simulation that allows for simulation directly from a covariance matrix or using PCA with an optional parameter for

Generate a correlation matrix and variance vector 2 ways:

1. Standard Pearson correlation/variance (you do not need to reimplement the cor() and var() functions).
2. Exponentially weighted $\lambda = 0.97$

Combine these to form 4 different covariance matrices. (Pearson correlation + var()), Pearson correlation + EW variance, etc.)

Simulate 25,000 draws from each covariance matrix using:

1. Direct Simulation
2. PCA with 100% explained
3. PCA with 75% explained
4. PCA with 50% explained

Calculate the covariance of the simulated values. Compare the simulated covariance to it's input matrix using the Frobenius Norm (L2 norm, sum of the square of the difference between the matrices). Compare the run times for each simulation.

What can we say about the trade offs between time to run and accuracy.

## Answer

We use all combinations of EW covariance, EW correlation, covariance, and correlation to get the four covariance matrices. As expected, I find that the PCA simulations are much faster than the direct simulations that use the Cholesky factorization directly. The PCA simulation with 100% variance explained is twice as fast as the direct simulation, and their precision is about the same. Their Frobenius norm does not vary too much, and the PCA simulation with 100% variance explained does not improve the precision by much. Thus, the PCA simulation with 100% variance explained is much more efficient and the precision is almost not affected.

For the PCA simulation, the running time decreases, but the precision also decreases as the variance explained decreases. From the graphs, we can see that the Frobenius norm increases by an order of magnitude when the variance explained decreases by 25%. It makes sense that if we use less information to do the simulation, we would not get the closest results, which means we lose some information to do a faster simulation.

Such a phenomenon is similar to problem 2. If we want to improve the accuracy, we need to do more computation to make each digital number as close as the original covariance matrix. If we put more time into the calculation, the accuracy will increase, and vice versa. Which method we choose depends on our scenario and our tasks.

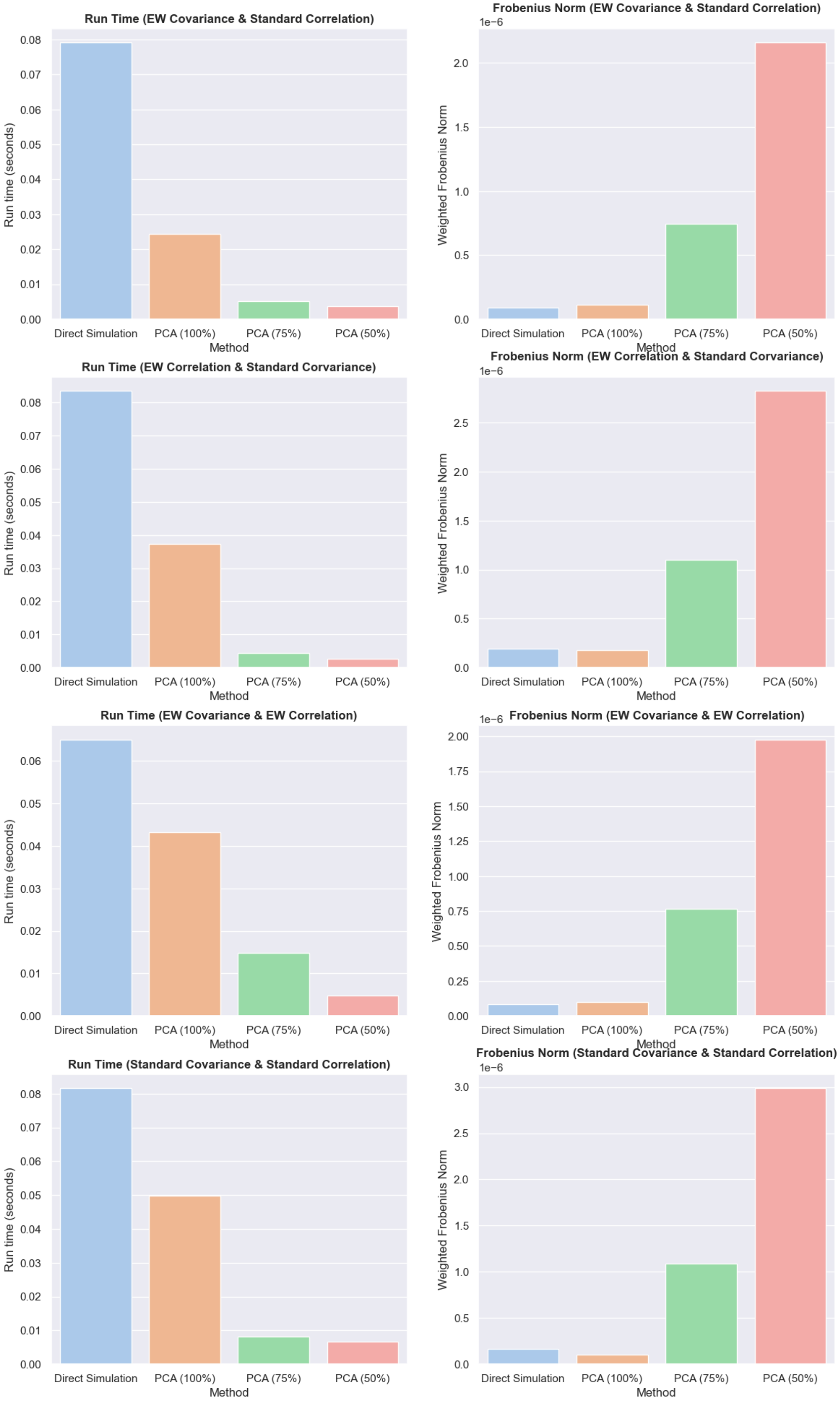
Therefore, if we want to do a simulation, we might choose to do a PCA simulation because it could reduce complexity, make the process faster, and maintain accuracy if the variance explained percentage is suitable.

**Table 1:** Run Time

| Variance | Direct Simulation | PCA (100%) | PCA (75%) | PCA (50%) |
|---|---|---|---|---|
| EW Covariance & Standard Correlation | 0.08814 | 0.04007 | 0.01278 | 0.00369 |
| EW Correlation & Standard Corvariance | 0.05663 | 0.03632 | 0.00991 | 0.0033 |
| EW Covariance & EW Correlation | 0.06585 | 0.02782 | 0.01198 | 0.00338 |
| Standard Covariance & Standard Correlation | 0.09259 | 0.02611 | 0.00479 | 0.00398 |

**Table 2:** Weighted Frobenius Norm

| Variance | Direct Simulation | PCA (100%) | PCA (75%) | PCA (50%) |
|---|---|---|---|---|
| EW Covariance & Standard Correlation | 8.40E-08 | 9.50E-08 | 7.44E-07 | 2.16E-06 |
| EW Correlation & Standard Corvariance | 1.50E-07 | 2.40E-07 | 1.10E-06 | 2.82E-06 |
| EW Covariance & EW Correlation | 9.20E-08 | 1.31E-07 | 7.72E-07 | 1.98E-06 |
| Standard Covariance & Standard Correlation | 1.52E-07 | 1.46E-07 | 1.08E-06 | 2.99E-06 |



Run Time (EW Covariance & Standard Correlation)
Run time (seconds)
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Frobenius Norm (EW Covariance & Standard Correlation)
1e−6
Weighted Frobenius Norm
0.0
0.5
1.0
1.5
2.0
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Run Time (EW Correlation & Standard Corvariance)
Run time (seconds)
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Frobenius Norm (EW Correlation & Standard Corvariance)
1e−6
Weighted Frobenius Norm
0.0
0.5
1.0
1.5
2.0
2.5
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Run Time (EW Covariance & EW Correlation)
Run time (seconds)
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Frobenius Norm (EW Covariance & EW Correlation)
1e−6
Weighted Frobenius Norm
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Run Time (Standard Covariance & Standard Correlation)
Run time (seconds)
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method
Frobenius Norm (Standard Covariance & Standard Correlation)
1e−6
Weighted Frobenius Norm
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Direct Simulation
PCA (100%)
PCA (75%)
PCA (50%)
Method