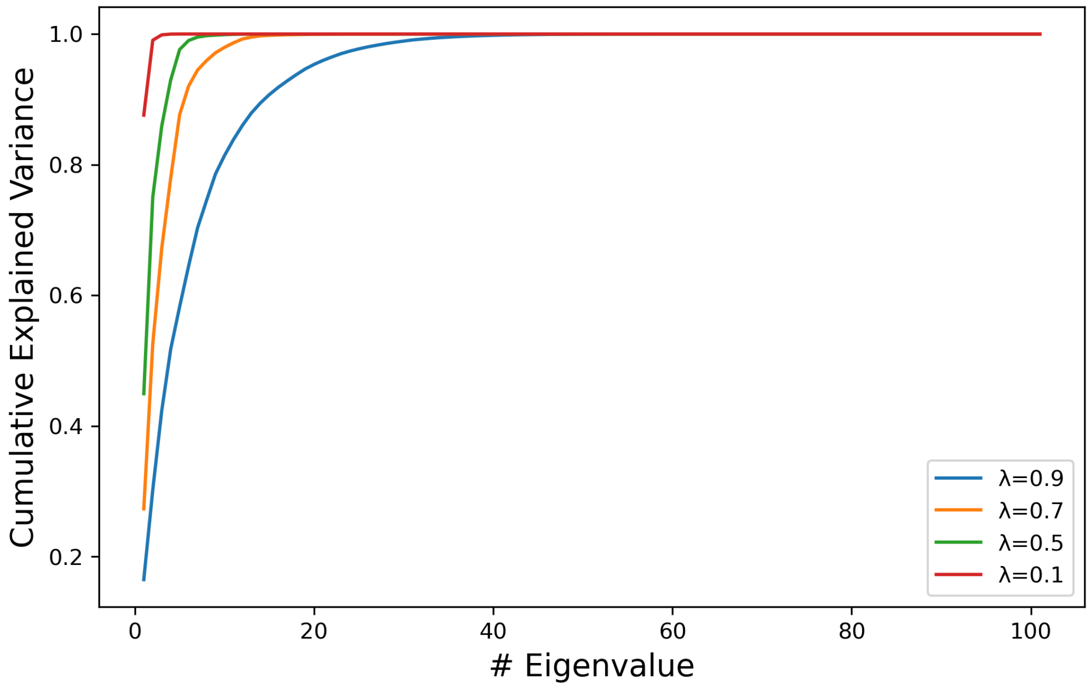
Problem 1

As we know about exponential weight, when λ becomes smaller, the greater weight will be assigned to recent data. The greater the λ is, the weights are more equally assigned to each time period. This explains what we see in the figure shown below- when λ is closer to 0, the fewer principal components will explain the variance of the exponentially weighted covariance matrix. For example, when λ=0.1, the first principal component alone explained over 80% of the variance, while the rest (other than the first 3) of the principal components barely explained any variance. On the other hand, when λ is closer to 1, we get more principal components that explained meaningful amount of variance.



Problem 2

```
#confirm that the output of near_psd is psd
matrix_p=near_psd(sigma,epsilon=0.0)
is_psd(matrix_p)
```

```
The out put is psd
```

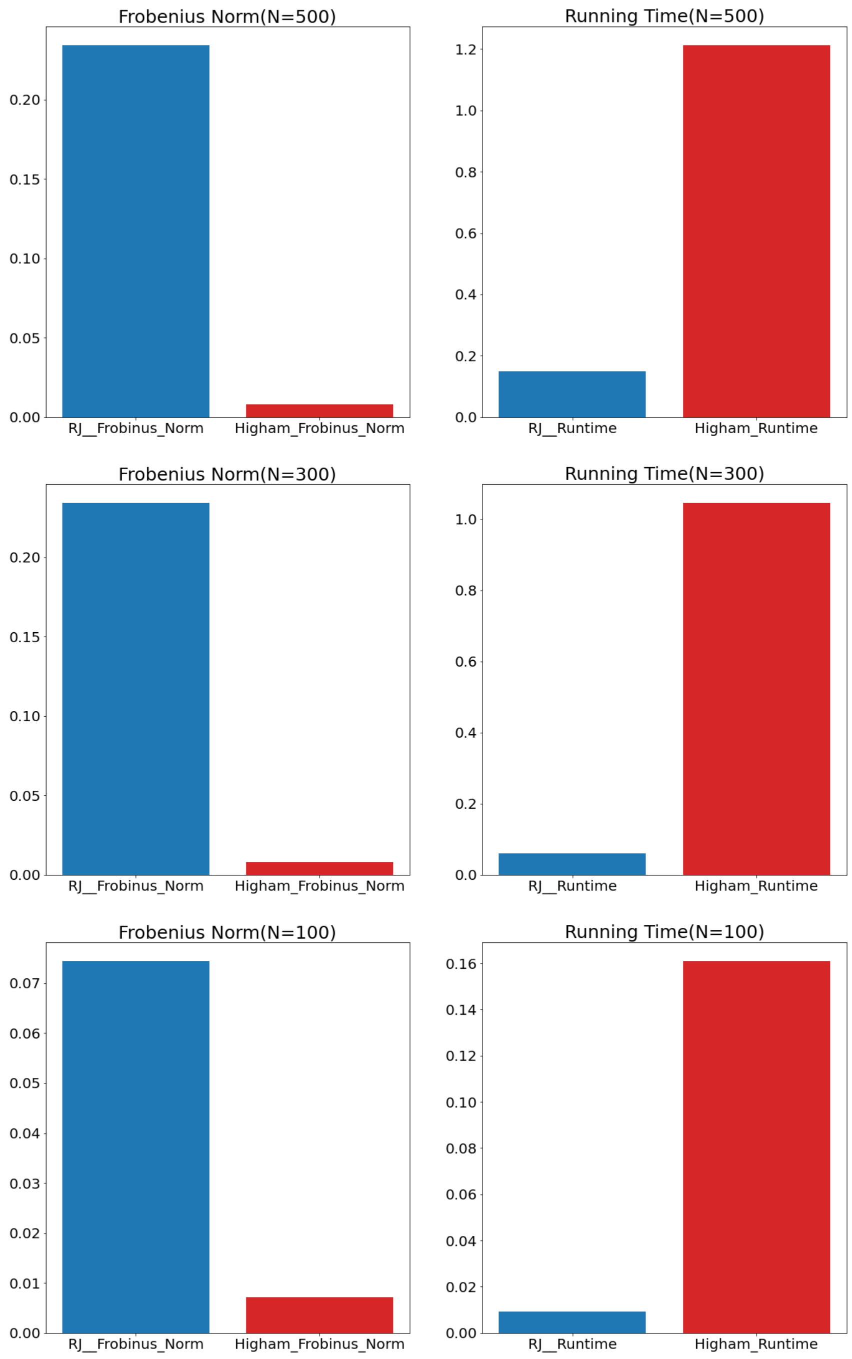
```
#confirm that the output o higham is psd
matrix_h=higham_nearestPSD(sigma)
is_psd(matrix_h)
```

```
The out put is psd
```

As the figure above shows, we confirmed that both of our outputs from near_psd() and Higham's method output PSD matrix, which means we fixed the matrix successfully.

The figure below shows the running time and Frobenius Norm for both methods when N=100,300,500. As this figure shows, the running time for Rebonato & Jackel's method

did not change a lot as N increase. The running time stay almost the same for N=500 and N=300, while decreased about 0.19 from N=300 to N=100. On the other hand, there are more obvious increase in running time for Higham's method as N increase.
At the same time, the Frobenius Norm for Higham's method stayed stable as N changes, but there are obvious increase on the Frobenius Norm for Higham's method as N increase.

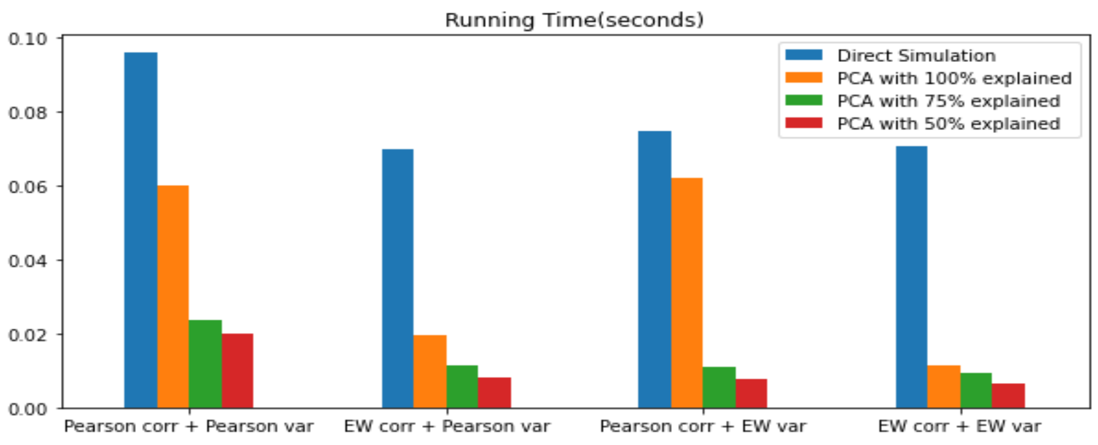


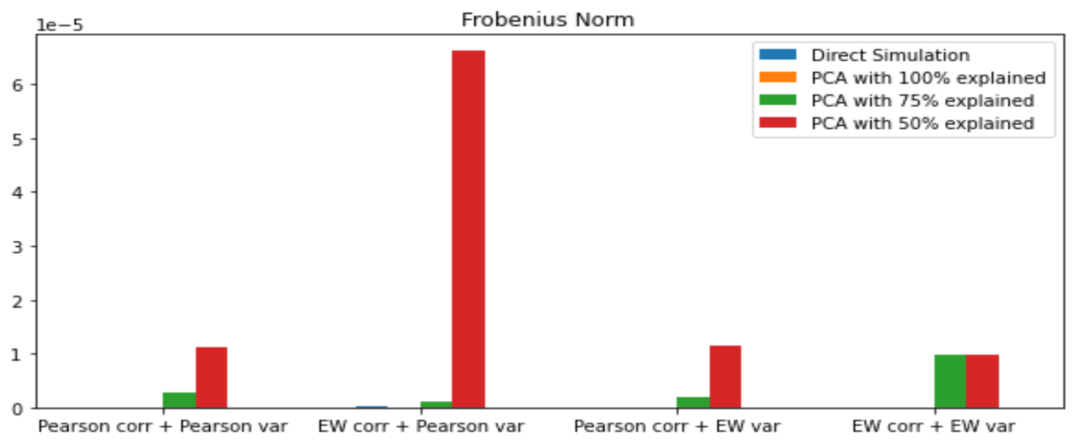
From above, we could say that the Rebonato & Jackel's method is fast but relatively inaccurate. While the Higham's method is accurate but relatively slow. There is not much difference between the two when handeling small matrix(ex. N=100). But we must make a choice between accuracy and running time when dealing with larger size of matrix.

Problem 3
(First table is for Running time and the Second table is for Frobenius Norm)

	Direct Simulation	PCA with 100% explained	PCA with 75% explained	PCA with 50% explained
Pearson corr + Pearson var	0.096083	0.060298	0.023698	0.020268
EW corr + Pearson var	0.070088	0.019543	0.011624	0.008474
Pearson corr + EW var	0.074882	0.062277	0.011024	0.007793
EW corr + EW var	0.070894	0.011634	0.009502	0.006530

	Direct Simulation	PCA with 100% explained	PCA with 75% explained	PCA with 50% explained
Pearson corr + Pearson var	4.180798e-08	4.944318e-08	0.000003	0.000011
EW corr + Pearson var	8.418738e-08	5.501477e-08	0.000001	0.000066
Pearson corr + EW var	4.703103e-08	3.109747e-08	0.000002	0.000011
EW corr + EW var	1.038263e-08	3.228382e-08	0.000010	0.000010





From the figures above, we noticed that generally, the direct simulation has the best accuracy and worst running time. The running time tends to decrease as the percentage explained by PCA decreases. The accuracy decreases as the percentage explained by PCA decreases. We also noticed that although we can be 5-10 times faster by using PCA with 50% and 75% variance explained, we will suffer from hundreds of times larger Frobenius Norms. Since the inaccuracy we get is too large, and the speed-up we get from it is relatively small, I doubt it is a good idea to use PCA with a low percentage explained. The key task is to maintain enough information while chasing a relatively optimal running time.