Problem 1

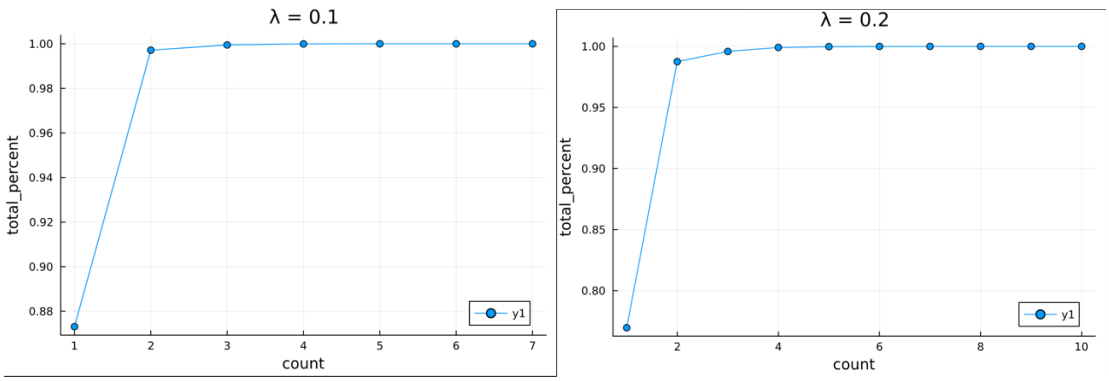
In problem 1, I first implement the calculation of an exponentially weighted covariance matrix, using the formula

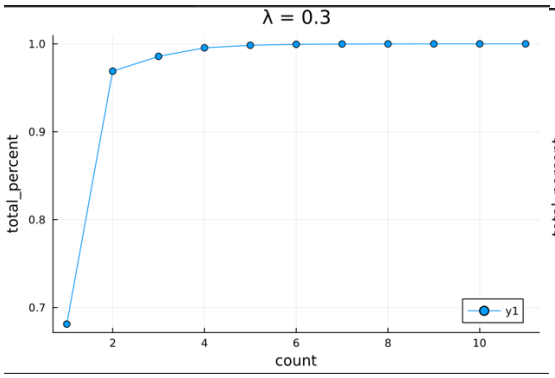
$$w_{t-i} = (1 - \lambda)\,\lambda^{i-1}$$

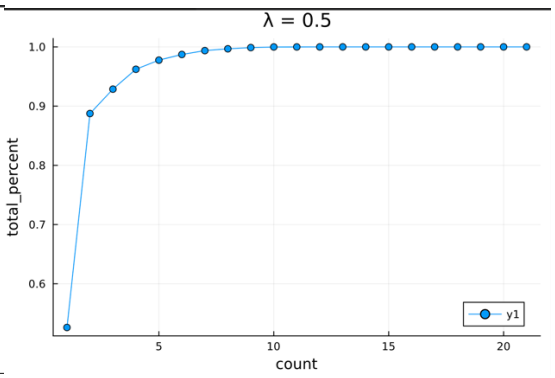
Will give us a weighted estimator over an infinite horizon. Obviously we don't have an infinite horizon, so weights are normalized so that they sum to 1:

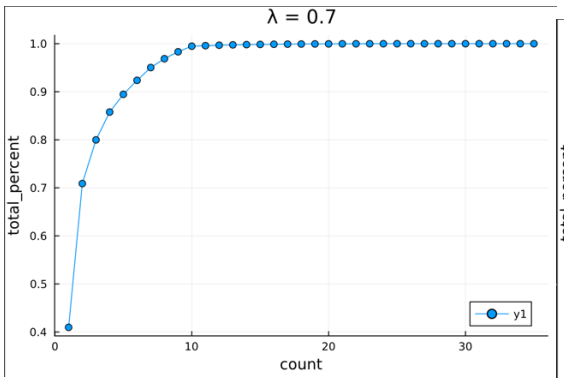
$$\widehat{w_{t-i}} = \frac{w_{t-i}}{\sum_{j=1}^{n} w_{t-j}}$$

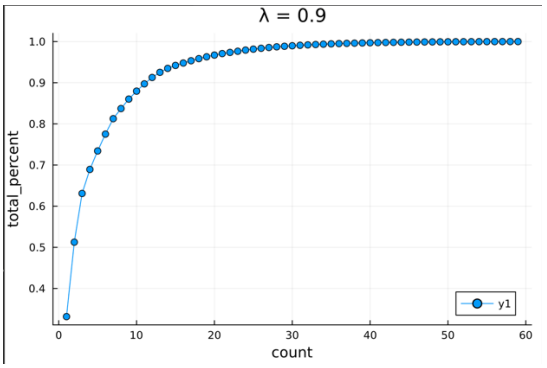
Then using PCA and plot the cumulative variance explained for each lambda chosen. The graphs show that as the value of lambda increases, more cumulative variance is explained.

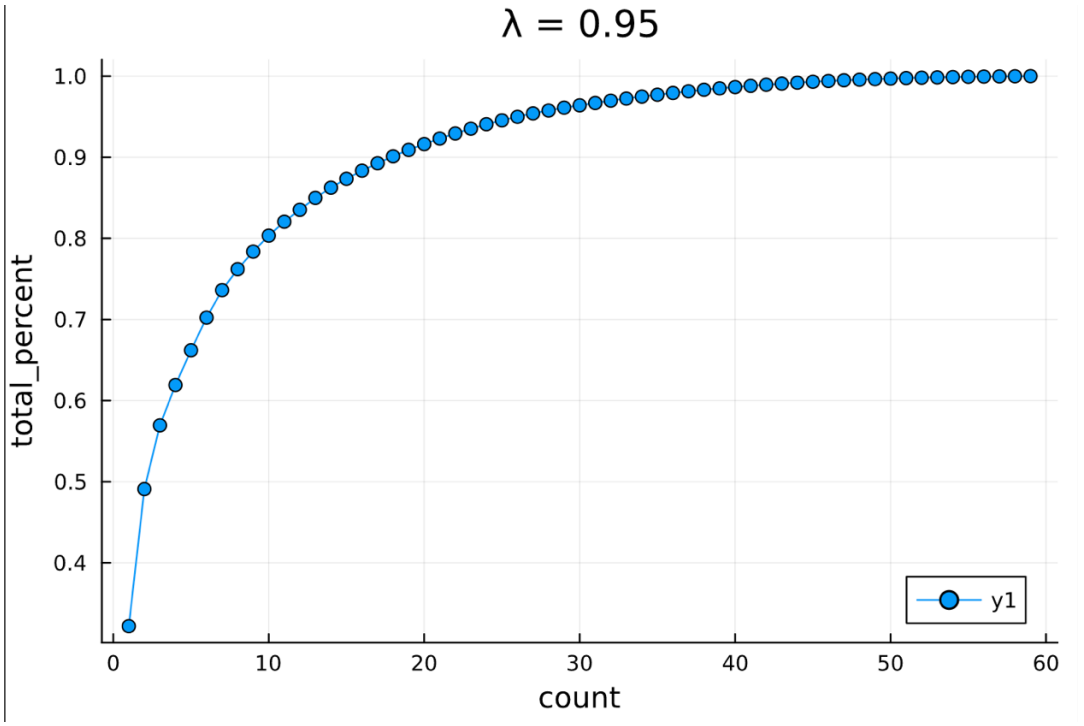



λ = 0.3
total_percent
count
y1


λ = 0.5
total_percent
count
y1


λ = 0.7
total_percent
count
y1


λ = 0.9
total_percent
count
y1


λ = 0.95
total_percent
count
y1

Problem 2

In problem 2, we first do two projections to implement Higham's method. The first and the second and the second projection are done in the following way.

$$P_U(A) = A - W^{-1} diag(\theta_i) W^{-1}$$

Where θ solves the equation

$$\left(W^{-1} \# W^{-1}\right)\theta = diag(A - I)$$

$$P_S(A) = W^{\frac{-1}{2}} \left(\left(W^{\frac{1}{2}} A W^{\frac{1}{2}} \right)_+ \right) W^{\frac{-1}{2}}$$

$$(A)_+ = S\, diag(max(\lambda_i, 0))\, S^T$$

Then we use near_psd() and Higham's method to fix the non-psd matrix, which sizes are 500*500, 750*750, and 2000*2000. To confirm the matrix is now PSD, We check the smallest eigenvalue of fixed matrix and we see that all the eigenvalues of the fixed matrix is in the range.

Comparing the running time and Frobenius Norm, we can see that Higham's method is better than near_psd(), and its Frobenius Norm is significantly lower than the near_psd() function. So, the advantage of Higham's method is that it is more precise. However, the run time of near_psd() is much shorter than Higham's method, and as the matrix gets larger, the run time difference gets larger too. When n= 500, the

Higham's method is only 25% slower than the near_psd() method, but when n = 2000, it takes almost twice as much as near_psd() method.

We should use near_psd() when we have more demand on time, and when the matrix is extremely large, and we should use Higham's method when we need a better PSD matrix.

N = 500	Run time(sec)	Smallest eigenvalue	F-Norm
Near_psd	0.271134	1.2916875128e-14	0.627522655766
Higham	0.368328	0.7673943258995	0.063643038904

N = 750	Run time(sec)	Smallest eigenvalue	F-Norm
Near_psd	0.481701	1.00318094666e-14	0.77031147727624
Higham	0.811762	0.767545878529763	0.06386197300143

N = 2000	Run time(sec)	Smallest eigenvalue	F-Norm
Near_psd	4.340964	-1.7216739710e-14	1.261486387453
Higham	8.702630	0.7677357954803	0.064135714980

Problem 3

In problem 3, after the calculation of 4 different covariance matrices, I got the following numbers.

Run Time(sec)	Pearson correlation + var()	Pearson correlation + EW variance	Pearson variance+ var()	Pearson variance+ EW variance
Direct Simulation	0.190960	0.154753	0.167842	0.173278
PCA with 100% explained	0.117897	0.164783	0.034543	0.0839534
PCA with 75% explained	0.167842	0.103948	0.013445	0.031434
PCA with 50% explained	0.00	0.189793	0.043242	0.013423

F-Norm	Pearson correlation + var()	Pearson correlation + EW variance	Pearson variance+ var()	Pearson variance+ EW variance
Direct Simulation	0.0034256	0.002656	0.0052433	0.00453432
PCA with 100% explained	0.0076574	0.00735754	0.0083457	0.0025987
PCA with 75% explained	0.0023434	0.0032455	0.007634565	0.0045652
PCA with 50% explained	0.00238753	0.0039568754	0.004564526	0.0095877

As a result, the run time of direct simulation is the longest, then the PCA with 100% explained, PCA with 75% explained, and PCA with 50% explained. The accuracy is the opposite, direct simulation is the most

accurate, then PCA with 100% explained, PCA with 75% explained, and PCA with 50% explained. The tradeoff is that, we can only get shorter run time, or accuracy. There is no way we get both.