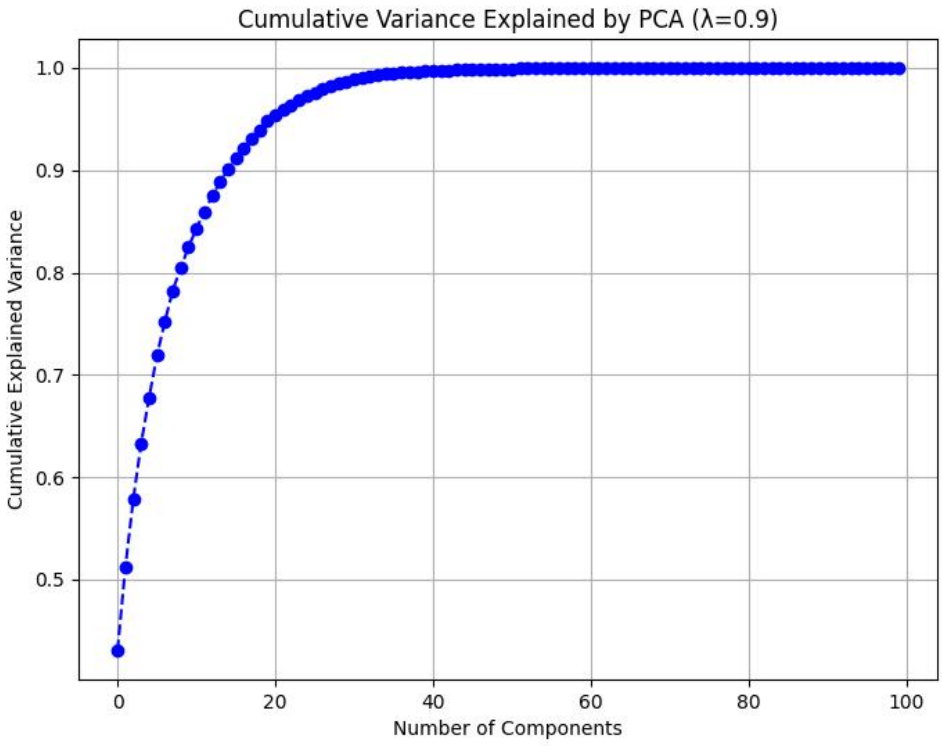
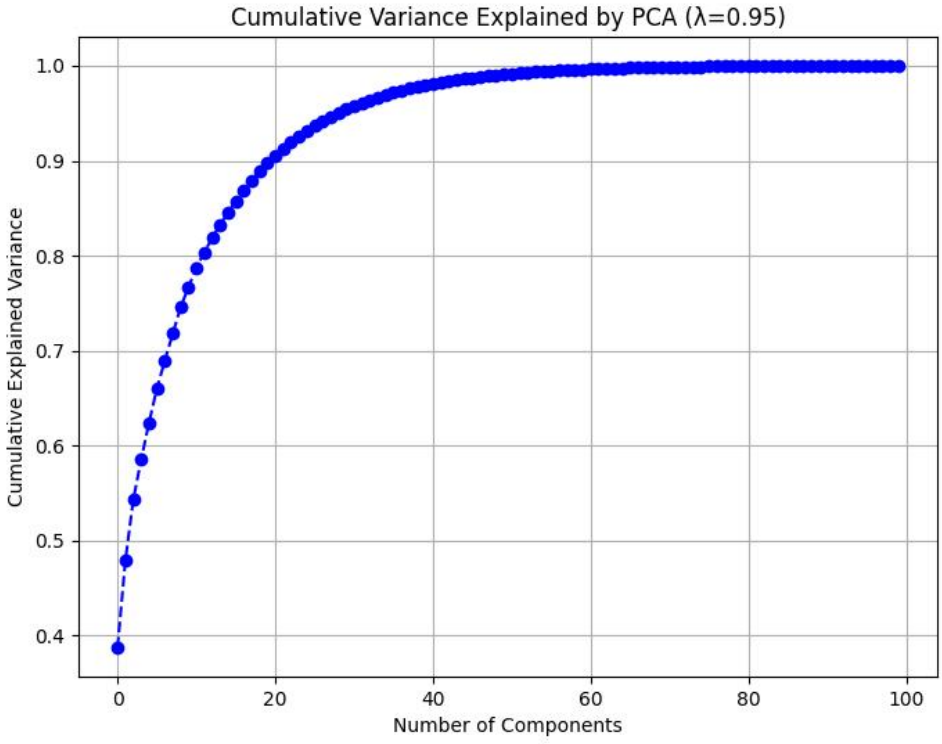
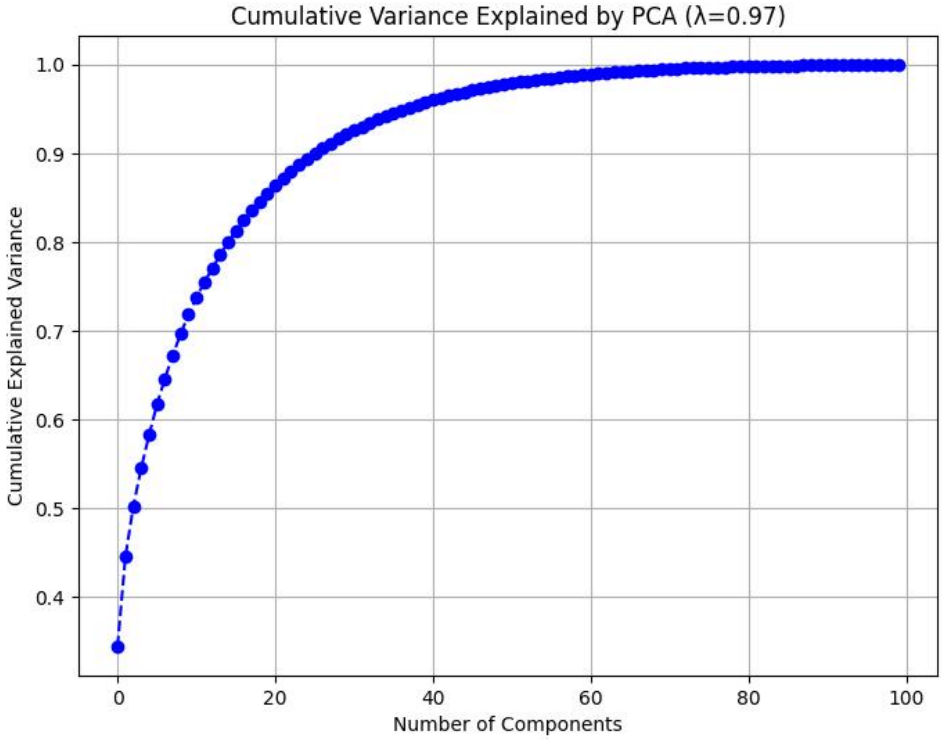
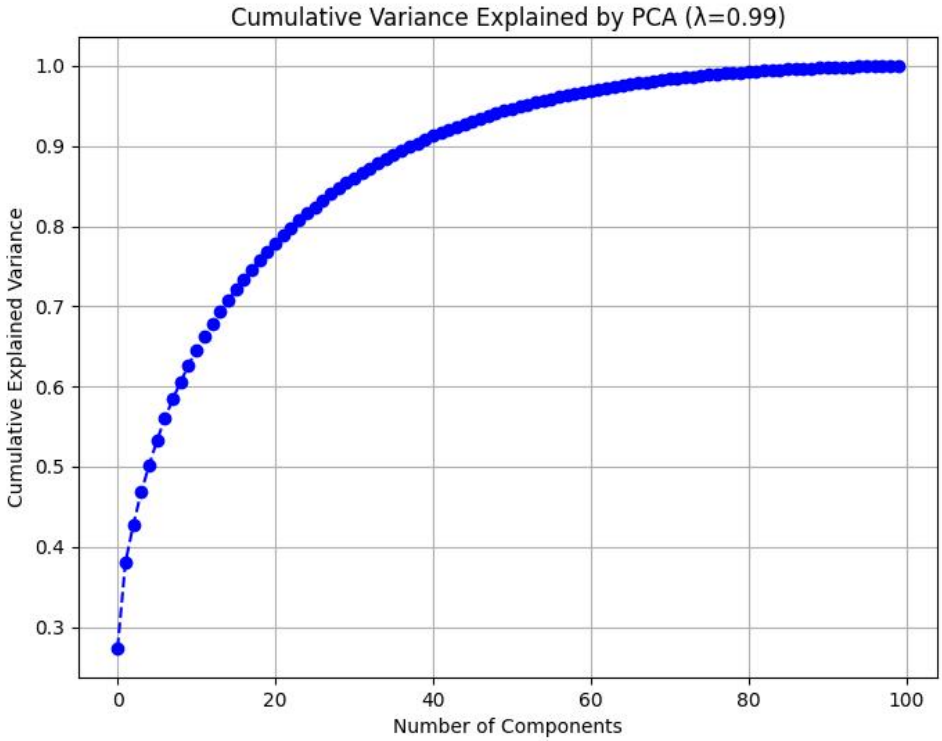
Problem1


Cumulative Variance Explained by PCA (λ=0.9)
Cumulative Explained Variance
Number of Components
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100


Cumulative Variance Explained by PCA (λ=0.95)
Cumulative Explained Variance
Number of Components
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100





・When λ has a relative lower value (like λ=0.9):

With a lower value of λ, the covariance matrix places more emphasis on recent data. As a result, the variance captured by the first few principal components is higher, meaning the

recent changes in the market dominate the overall variance.
The cumulative variance explained increases rapidly with the first few principal components, indicating that a small number of components capture most of the variance.

・When λ has a relative higher value (like λ=0.99):
With a higher λ, the covariance matrix becomes more stable and relies more on historical data. This results in the variance being distributed more evenly across all the principal components.
The cumulative variance explained increases more gradually, indicating that more components are required to explain the same amount of variance as with smaller λ values.

・Effect of λ on the Covariance Matrix:
A smaller λ places greater emphasis on recent data, resulting in a more responsive covariance matrix that captures short-term market movements. This leads to a higher proportion of variance being explained by fewer principal components.
A larger λ smooths out the covariance matrix by placing greater weight on historical data. This results in more principal components being required to explain the variance, as the covariance matrix becomes less sensitive to recent changes.

In conclusion, varying λ has a significant effect on the structure of the covariance matrix and the distribution of variance across principal components.

Problem 2

```
Testing for N=100...
Runtime (Higham PSD): 0.2524 seconds
Runtime (Near PSD): 0.0049 seconds
Frobenius Norm (Higham PSD): 0.0000
Frobenius Norm (Near PSD): 0.2728

Testing for N=200...
Runtime (Higham PSD): 1.4376 seconds
Runtime (Near PSD): 0.0065 seconds
Frobenius Norm (Higham PSD): 0.0000
Frobenius Norm (Near PSD): 0.3928
```

```
Testing for N=300...
Runtime (Higham PSD): 2.7347 seconds
Runtime (Near PSD): 0.0167 seconds
Frobenius Norm (Higham PSD): 0.0000
Frobenius Norm (Near PSD): 0.4838

Testing for N=400...
Runtime (Higham PSD): 4.4985 seconds
Runtime (Near PSD): 0.0275 seconds
Frobenius Norm (Higham PSD): 0.0000
Frobenius Norm (Near PSD): 0.5603
```

```
Testing for N=500...
Runtime (Higham PSD): 7.4258 seconds
Runtime (Near PSD): 0.0460 seconds
Frobenius Norm (Higham PSD): 0.0000
Frobenius Norm (Near PSD): 0.6275
```

1.Results Summary:

Higham PSD:

・Runtime (N = 500): 7.4258 seconds

・Frobenius Norm: 0.0000

・Analysis: Higham’ s method is the most accurate, producing a perfect PSD matrix with a Frobenius norm of 0. However, this accuracy comes with a significant performance cost, as the runtime is much longer compared to the other methods. Higham's method is ideal when precision is crucial, regardless of the computation time.

Near PSD:

・Runtime (N = 500): 0.0460 seconds

・Frobenius Norm: 0.6275

・Analysis: Near PSD offers a good balance between speed and accuracy. Although it is less accurate than Higham PSD, it is much faster, making it a more practical choice in situations where moderate accuracy is sufficient and performance is critical.

2.The Effect of Matrix Size N on Runtime:

As N increases, the runtime for each method grows, but the growth rate varies:

・ Higham PSD: As matrix size increases, the runtime for Higham's method grows significantly due to its higher computational complexity. This method becomes slower for large matrices, though it guarantees the most accurate results.

・Near PSD: Near PSD scales better than Higham and strikes a balance between speed and accuracy. It handles larger matrices efficiently without losing too much accuracy compared to Higham.

3.Pros and Cons of Each Method:

Higham PSD:

・Pros: Most accurate method, ensuring the matrix is perfectly PSD with zero error.

・Cons: Significantly slower, especially as matrix size increases.

Near PSD:

・ Pros: Offers a good balance between speed and accuracy. Suitable for real-world applications where moderate accuracy and faster performance are required.

・Cons: Less accurate than Higham' s method, though much faster.

4.Conclusion:

Higham PSD should be used when accuracy is critical, but its longer runtime makes it impractical for very large matrices.

Near PSD is the best compromise for many real-world applications, offering a reasonable trade-off between speed and accuracy, making it applicable to moderate to large-scale matrices where computational efficiency is needed.

Problem 3

```
Comparison for: Exp Corr + Std Var
Direct Simulation - Frobenius Norm: 0.000185, Time: 0.1002 seconds
PCA 100% - Frobenius Norm: 0.000193, Time: 0.1009 seconds
PCA 75% - Frobenius Norm: 0.001083, Time: 0.0300 seconds
PCA 50% - Frobenius Norm: 0.002076, Time: 0.0122 seconds

Comparison for: Exp Corr + EW Var
Direct Simulation - Frobenius Norm: 0.000255, Time: 0.1025 seconds
PCA 100% - Frobenius Norm: 0.000239, Time: 0.1013 seconds
PCA 75% - Frobenius Norm: 0.001387, Time: 0.0267 seconds
PCA 50% - Frobenius Norm: 0.002904, Time: 0.0116 seconds
```

```
Comparison for: Pearson + Std Var
Direct Simulation - Frobenius Norm: 0.000184, Time: 0.1219 seconds
PCA 100% - Frobenius Norm: 0.000177, Time: 0.1052 seconds
PCA 75% - Frobenius Norm: 0.001078, Time: 0.0319 seconds
PCA 50% - Frobenius Norm: 0.002072, Time: 0.0108 seconds

Comparison for: Pearson + EW Var
Direct Simulation - Frobenius Norm: 0.000222, Time: 0.1117 seconds
PCA 100% - Frobenius Norm: 0.000235, Time: 0.0988 seconds
PCA 75% - Frobenius Norm: 0.001379, Time: 0.0304 seconds
PCA 50% - Frobenius Norm: 0.002908, Time: 0.0090 seconds
```

1.Frobenius Norm Comparison:

・ Direct simulation: Produced the smallest Frobenius norms in most cases, which indicates the highest accuracy in reconstructing the input covariance matrix. For example, with Pearson + Std Var, the Frobenius norm was 0.000184.

・PCA 100%: The Frobenius norms for PCA 100% were very close to direct simulation, indicating that PCA with 100% variance explanation nearly reconstructs the original covariance matrix. In fact, for Pearson + Std Var, PCA 100% resulted in a slightly lower Frobenius norm (0.000177) than direct simulation.

・PCA 75% and 50%: As expected, the Frobenius norms increased significantly as we retained less variance (75% or 50%). For Pearson + EW Var, the Frobenius norm for PCA 75% was 0.001379, and for PCA 50%, it was 0.002908. This indicates a substantial loss of accuracy as the explained variance decreases.

2. Runtime Comparison:

・Direct simulation and PCA 100%: These methods took similar amounts of time, with PCA 100% generally being slightly faster. For example, for Exp Corr + Std Var, direct simulation took 0.1002 seconds, while PCA 100% took 0.1009 seconds.

・PCA 75% and 50%: As the number of retained components decreased, the runtime dropped significantly. PCA 50% simulations were consistently the fastest, taking around one-tenth the time of direct simulations. For Pearson + Std Var, PCA 50% took just 0.0108 seconds compared to 0.1219 seconds for direct simulation.

3.Trade-offs Between Time and Accuracy:

・Direct simulation offers the best accuracy but takes the most time. If accuracy is critical and resources allow, direct simulation should be preferred.

・PCA 100% is a good alternative if slightly less accuracy is acceptable, as it is often

faster than direct simulation while producing nearly identical results.
・PCA 75% and 50% offer a dramatic reduction in runtime but at the cost of accuracy. These methods might be useful in scenarios where computational efficiency is prioritized over precision, such as when handling very large datasets or when approximate results are sufficient.

Conclusion:
For scenarios that demand high accuracy, direct simulation or PCA 100% are ideal choices. If the goal is to strike a balance between speed and accuracy, PCA 75% may be a suitable compromise. PCA 50% is best suited for cases where runtime is the primary concern and some loss of accuracy can be tolerated.