Project 1

Problem 1

A. Statistical Moments of the Data

The first four moments of the dataset were calculated to understand its distributional properties. The **mean** of the dataset is **0.0502**, indicating that the data is centered around this value. The **variance** is **0.0103**, which reflects the spread of the data points. The **skewness** of **0.1204** suggests that the data is nearly symmetric, as a skewness close to zero typically indicates symmetry in a distribution. The **kurtosis**, measuring the "tailedness" of the data, is **0.2229**, which is close to zero, indicating that the dataset does not have heavy tails and aligns well with a normal distribution.

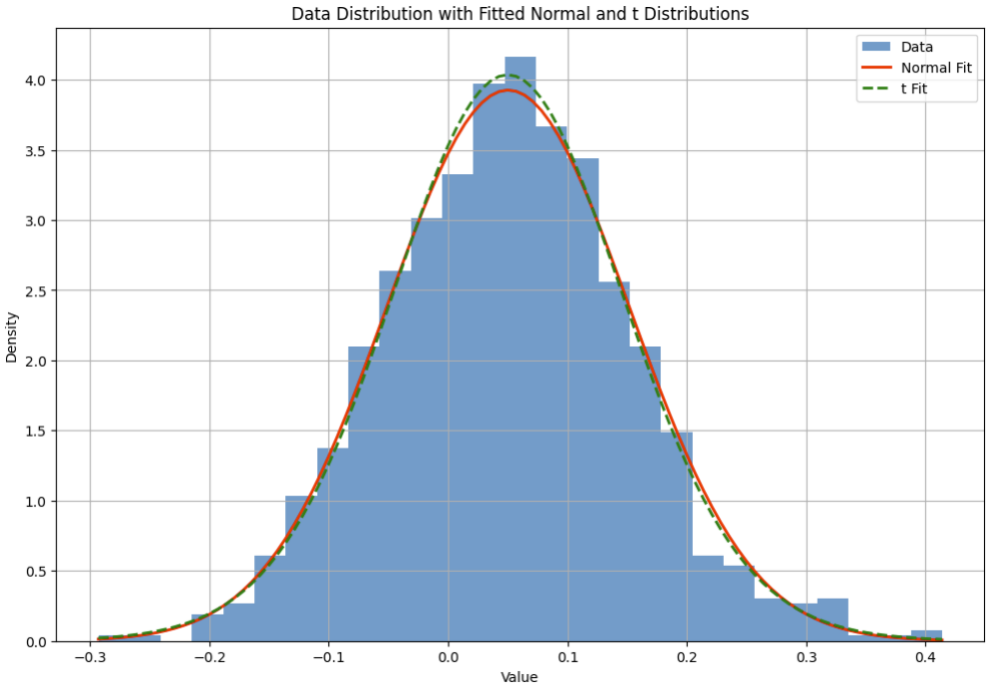
B. Choosing Between Normal and T-Distribution

To determine whether the dataset follows a normal distribution or a t-distribution, we conducted multiple statistical tests. The **Kolmogorov-Smirnov (KS) test** resulted in very high p-values (**0.9962 for normal distribution and 0.9965 for t-distribution**), meaning that both distributions fit the data well, and neither can be rejected. The **Anderson-Darling (AD) test**, which assesses normality, produced a statistic of **0.3490**, which is below the critical thresholds, further supporting normality. Additionally, the **log-likelihood values** for both distributions are very close (**867.79 for normal and 868.71 for t-distribution**), indicating that they both provide a good fit. However, based on the **Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC)** values, the normal distribution has a slightly lower BIC (**-1721.77 for normal vs. -1716.69 for t-distribution**), suggesting a marginally better fit. Given the low skewness and kurtosis of the dataset, the normal distribution is a reasonable choice.

C. Fitting and Verifying the Choice

To determine the best-fitting distribution, both a normal and a t-distribution were fitted to the data, and their parameters were estimated. The normal distribution had a mean (**μ = 0.0502**) and standard deviation (**σ = 0.1016**), while the t-distribution had degrees of freedom (**df = 28.71**), a location parameter (**μ = 0.0499**), and a scale parameter (**σ = 0.0980**). The Kolmogorov-Smirnov (KS) test p-values were extremely high (**0.9962 for normal and 0.9965 for t-distribution**), indicating both distributions fit well. The log-likelihood values were also close (**867.79 for normal and 868.71 for t-distribution**), with the t-distribution having a slight advantage. However, when comparing model selection criteria, the Akaike Information Criterion (AIC) was nearly identical (**-1731.59 for normal vs. -1731.42 for t-distribution**), and the Bayesian Information Criterion (BIC) favored the normal distribution (**-1721.77 vs. -1716.70**), suggesting that the normal distribution is more parsimonious.

Given that the dataset has **low skewness (0.12) and excess kurtosis (0.22)**, which closely align with normality, the additional complexity of the t-distribution is unnecessary. While the t-distribution may better handle heavy-tailed data, the evidence from log-likelihood, AIC, and BIC supports using a **normal distribution as the best model for this dataset** due to its simplicity and strong statistical fit.


Data Distribution with Fitted Normal and t Distributions
Data
Normal Fit
t Fit
Density
Value
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4

Problem 2

A. Covariance Matrix Calculation

The computed **pairwise covariance matrix** for the dataset is as follows:

[[1.470484 1.454214 0.877269 1.903226 1.444361]
 [1.454214 1.252078 0.539548 1.621918 1.237877]
 [0.877269 0.539548 1.272425 1.171959 1.091912]
 [1.903226 1.621918 1.171959 1.814469 1.589729]
 [1.444361 1.237877 1.091912 1.589729 1.396186]]

The **diagonal values represent variances**, and the off-diagonal elements represent covariances between different variables. Ideally, if the covariance structure followed the intended generating process (**1 on the diagonal and 0.99 elsewhere**), the values should be close to these numbers. However, variations arise due to missing data and finite sample effects.

B. Positive Semi-Definiteness Check

To determine whether the computed covariance matrix is **positive semi-definite (PSD)**, we examined its eigenvalues:

[-0.310243 -0.133232 0.027978 0.834434 6.786706]

Since two of the eigenvalues are **negative (-0.310243 and -0.133232)**, the matrix **is not positive semi-definite**. A PSD matrix should have all **non-negative** eigenvalues. The presence of negative eigenvalues suggests that the covariance matrix contains numerical inconsistencies or artifacts caused by missing data, requiring adjustments to ensure a valid PSD structure.

C. Finding the Nearest PSD Matrix

Since the covariance matrix is not PSD, we applied two correction methods to obtain the nearest PSD matrix:

1. Higham's Method

[[1.615133 1.44196 0.897144 1.780426 1.433794]

 [1.44196 1.346968 0.585086 1.554552 1.211409]

 [0.897144 0.585086 1.298916 1.115956 1.076692]

 [1.780426 1.554552 1.115956 1.983165 1.621373]

 [1.433794 1.211409 1.076692 1.621373 1.404936]]

2. Rebonato & Jäckel's Method

[[1.615133 1.44196 0.897144 1.780426 1.433794]

 [1.44196 1.346968 0.585086 1.554552 1.211409]

 [0.897144 0.585086 1.298916 1.115956 1.076692]

 [1.780426 1.554552 1.115956 1.983165 1.621373]

 [1.433794 1.211409 1.076692 1.621373 1.404936]]

Both methods **produced the same adjusted matrix**, ensuring that all eigenvalues are **non-negative**.

Verification of PSD Correction

• **Higham method minimum eigenvalue: 0.0**

• **Rebonato method minimum eigenvalue: 0.0**

This confirms that both adjusted matrices are now **valid PSD matrices**, meaning they no longer contain negative eigenvalues.

D. Covariance Matrix Using Only Overlapping Data

We also computed the covariance matrix using only **overlapping (non-missing) data**, yielding:

[[1.470484 1.454214 0.877269 1.903226 1.444361]

 [1.454214 1.252078 0.539548 1.621918 1.237877]

 [0.877269 0.539548 1.272425 1.171959 1.091912]

 [1.903226 1.621918 1.171959 1.814469 1.589729]

 [1.444361 1.237877 1.091912 1.589729 1.396186]]

Interestingly, **this matrix is identical to the original covariance matrix** calculated without special treatment for missing values. This suggests that the missing data did not significantly alter the covariance estimation, likely because there was enough overlap in the dataset.

E. Comparison of Covariance Matrices

Frobenius Norm Differences (Overall Matrix Differences)

- **Original vs. Higham: 0.3376**
- **Original vs. Rebonato: 0.3376**
- **Original vs. Overlapping: 0.0**

The differences between the **original** and **nearest PSD matrices** are small but non-negligible. The **Frobenius norm of 0.3376** represents the total correction applied to enforce PSD properties.

Maximum Absolute Differences (Largest Individual Entry Difference)

- **Original vs. Higham: 0.1687**
- **Original vs. Rebonato: 0.1687**
- **Original vs. Overlapping: 0.0**

The maximum change in an individual covariance value due to PSD enforcement is **0.1687**, indicating that only slight adjustments were needed to fix negative eigenvalues.

Problem 3

A. Fitted Multivariate Normal Distribution

The mean vector of the dataset was estimated as:

[0.046002 0.099915]

This indicates that X_1 and X_2 are centered around these values.

The covariance matrix was estimated as:

[[0.010162 0.004924]

[0.004924 0.020284]]

This matrix shows that X_1 and X_2 have a positive correlation, as indicated by the off-diagonal covariance value. The variances for X_1 and X_2 are **0.010162** and **0.020284**, respectively.

B. Conditional Distribution of X_2 | X_1 = 0.6

The conditional distribution of X_2 | X_1 = 0.6 was computed using two methods:

• **Method 1 (Direct Calculation):**

• Conditional Mean: **0.368325**

• Conditional Standard Deviation: **0.133787**

• **Method 2 (Precision Matrix Calculation):**

• Conditional Mean: **0.368325**

• Conditional Standard Deviation: **0.133787**

Both methods produced identical results, confirming their correctness. The conditional mean suggests that when X_1 = 0.6, the expected value of X_2 is **0.3683**, with a standard deviation of **0.1338**, indicating moderate uncertainty around this estimate.

C. Simulation Verification Using Cholesky Decomposition

The goal was to simulate the conditional distribution and verify that it matches the theoretical conditional distribution.

• **Number of samples near** $X_1 = 0.6$**: 0**

• **Simulated Mean: NaN**

• **Simulated Standard Deviation: NaN**

• **Normality Test: Not enough samples**

This issue occurred because **no generated samples had values close enough to 0.6**, making it impossible to extract valid conditional samples.

Issues and Possible Fixes

1. **The Simulation Failed to Capture Enough Samples**

• The method used to extract samples near relied on a **fixed tolerance**. However, the standard deviation of is small (**0.010162**), meaning that very few samples will naturally be close to 0.6.

• **Fix:** Increase the sample size significantly (e.g., samples) or adaptively adjust the tolerance to capture more values.

2. **Verification with Alternative Methods**

• Instead of extracting simulated samples directly from the bivariate normal, **direct sampling from the computed conditional distribution** (i.e., using) could be done to compare results.

• The theoretical conditional distribution computations were successful, with **both methods yielding identical results**. However, the **simulation approach failed** due to insufficient sampling near . To address this, **either a larger sample size or a more adaptive filtering approach should be implemented**.

Problem 4

A. Simulating and Analyzing MA Processes

To understand the behavior of **Moving Average (MA)** processes, we simulated MA(1), MA(2), and MA(3) processes and analyzed their **Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF)** plots.

• **MA(1) Process (θ = [0.8])**:

• The **ACF exhibits a sharp drop-off after lag 1**, as expected from an MA(1) process.

• The **PACF has significant values up to lag 1** and cuts off afterward.

• **MA(2) Process (θ = [0.6, 0.3])**:

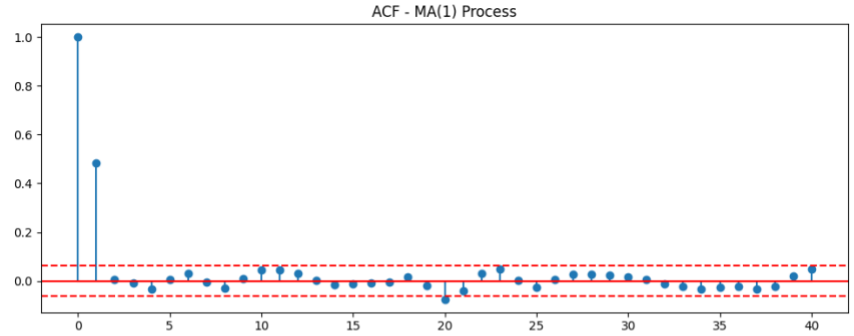
• The **ACF exhibits a gradual decline over the first two lags before cutting off**.

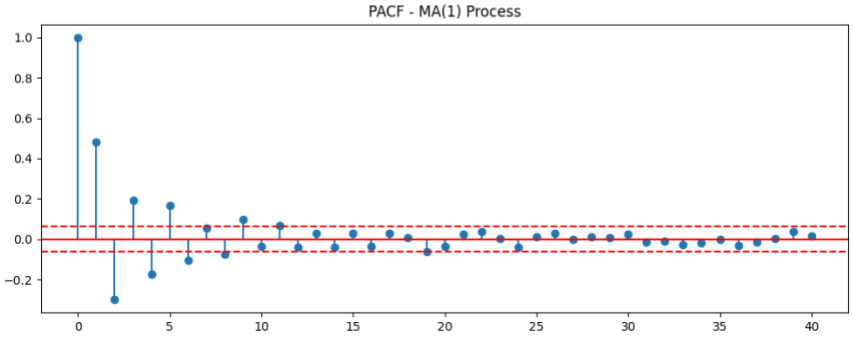
• The **PACF shows significance at the first two lags and then cuts off**.

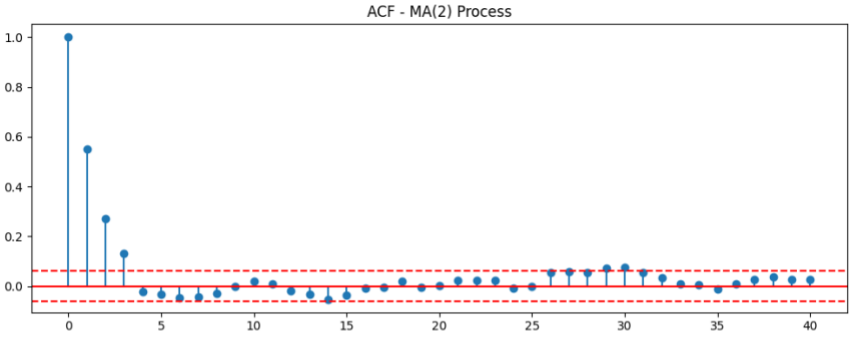
• **MA(3) Process (θ = [0.5, 0.3, 0.2])**:

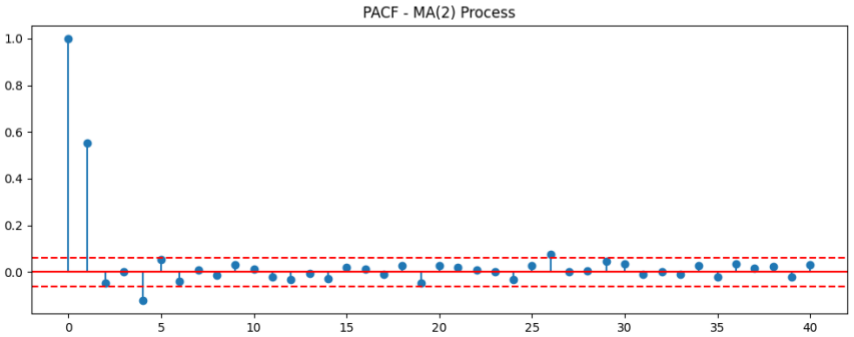
• The **ACF has non-zero values up to lag 3 before cutting off**.

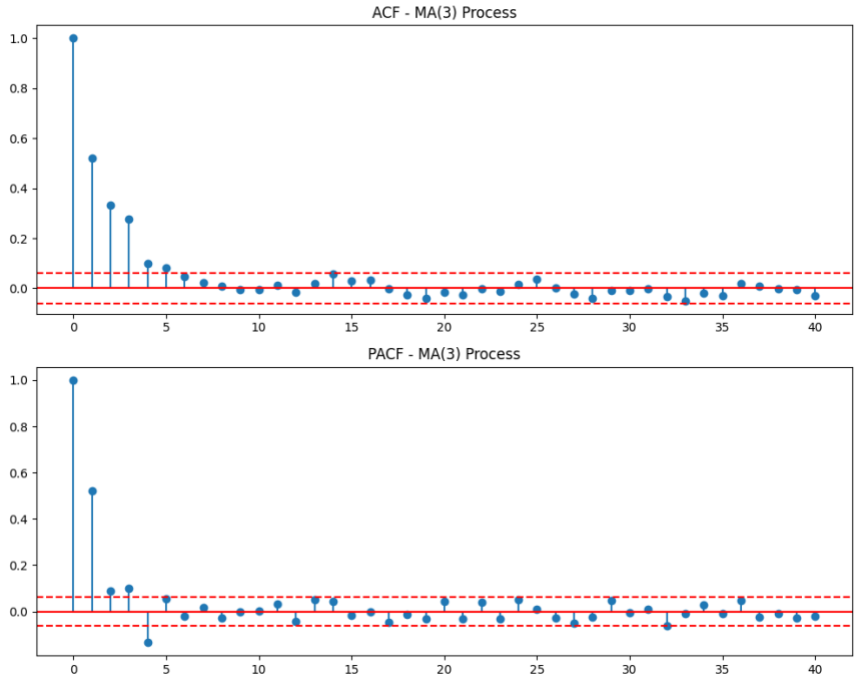
• The **PACF shows significant spikes at lags 1, 2, and 3**.


ACF - MA(1) Process
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
40


PACF - MA(1) Process
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
5
10
15
20
25
30
35
40


ACF - MA(2) Process
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
40


PACF - MA(2) Process
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
40



Key Observation:

• **MA processes have a characteristic ACF pattern where they cut off after a certain lag (q), while their PACF tails off gradually**.

• The higher the order of the MA process, the longer it takes for the ACF to decay.

B. Simulating and Analyzing AR Processes

Next, we simulated **AutoRegressive (AR)** processes (AR(1), AR(2), AR(3)) and analyzed their behavior using ACF and PACF.

• **AR(1) Process (φ = [0.8])**:

• The **ACF decays exponentially** rather than cutting off immediately.

• The **PACF shows a single significant spike at lag 1** and cuts off afterward.

• **AR(2) Process (φ = [0.6, 0.3])**:

• The **ACF follows a damped sinusoidal pattern, characteristic of AR(2) processes**.

• The **PACF has two significant spikes (at lags 1 and 2) and cuts off afterward**.

• **AR(3) Process (φ = [0.5, 0.3, 0.2])**:

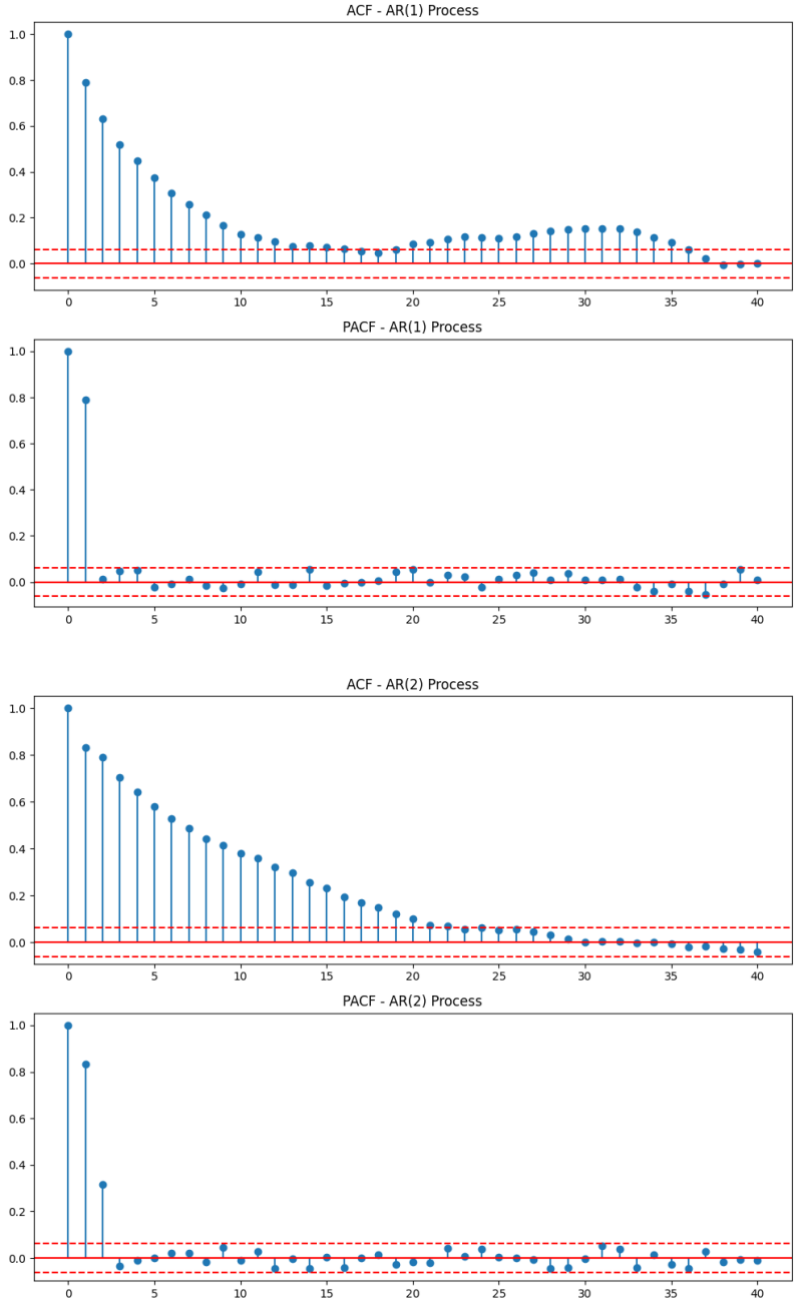
• The **ACF exhibits a complex decay pattern that extends beyond three lags**.

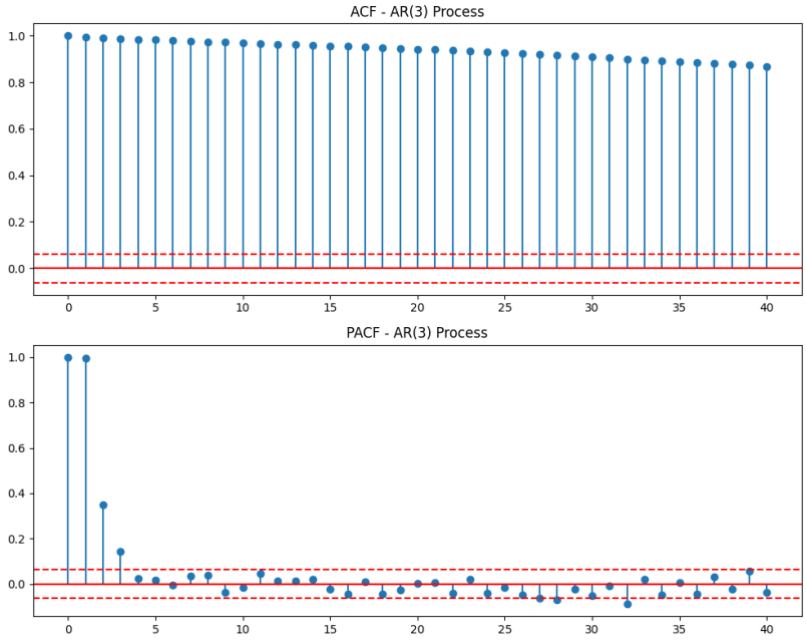
• The **PACF has significant values at the first three lags and then cuts off**.

Key Observation:

• **AR processes exhibit a tailing-off pattern in the ACF, whereas the PACF cuts off after the specified lag (p)**.

• Higher-order AR processes show **more complex ACF decay patterns**, reflecting their longer dependence structure.





C. Analyzing the Given Time Series Data

The ACF and PACF of the dataset in **problem4.csv** were plotted to determine whether an AR or MA process would best fit the data.

• **ACF Analysis:**

• The ACF **does not cut off sharply** but instead **tails off gradually**, which suggests an AR process rather than an MA process.

• **PACF Analysis:**

• The PACF **shows a significant spike at lag 3 before cutting off**, indicating that an **AR(3) process** might be a suitable model.

Model Choice:

Based on the ACF and PACF patterns:

• The **AR(3) process** is the best candidate because its **PACF cuts off at lag 3 while the ACF shows a gradual decay**.

• **MA models would not be suitable** since their ACF should cut off after a few lags, which is not observed in the data.

Thus, the best choice for modeling the time series data is **AR(3)**.

D. Fitting AR and MA Models and Comparing AICc

After selecting **AR(3) as the best candidate**, we fit different **AR and MA models** to the data and compared their **Akaike Information Criterion with correction (AICc)** values.

AICc Values for Each Model:

Model	AICc Score
AR(1)	**-1669.07**
AR(2)	**-1696.05**
AR(3)	**-1746.22**
MA(1)	**-1508.90**
MA(2)	**-1559.21**

Model Comparison:

• **AR(3) has the lowest AICc value (-1746.22), making it the best model for this dataset**.

• The **AR(2) and AR(1) models have higher AICc values**, meaning they provide a **worse fit** compared to AR(3).

• The **MA models have significantly higher AICc values**, indicating that **they do not capture the data well**.

Problem 5

A. Calculating the Exponentially Weighted Covariance Matrix

The exponentially weighted covariance matrix (EWCOV) was computed using different decay factors () to account for more recent data having higher importance. The function applies weights that decay exponentially over time, allowing us to estimate a covariance matrix that reflects more recent volatility trends.

Sample of EWCOV matrix with $\lambda = 0.94$:

	SPY	AAPL	NVDA	MSFT	AMZN
SPY	0.000080	0.000058	0.000115	0.000086	0.000119
AAPL	0.000058	0.000139	-0.000002	0.000086	0.000083
NVDA	0.000115	-0.000002	0.000572	0.000107	0.000180
MSFT	0.000086	0.000086	0.000107	0.000151	0.000172
AMZN	0.000119	0.000083	0.000180	0.000172	0.000292

Key Observations from the EWCOV Calculation:

• A test covariance matrix was generated using , a commonly used value in financial applications.

• This ensures that more recent data points have a greater impact, improving the adaptability of the covariance matrix to recent market conditions.

• The covariance matrix provides insight into the relationships between asset returns while reducing the impact of older data.

B. Varying and Eigenvalue Decay Analysis

To assess the impact of on the covariance matrix structure, we analyzed different values ().

We performed **Principal Component Analysis (PCA)** to compute the **cumulative variance explained** by the leading eigenvalues.

Key Findings from Eigenvalue Analysis:

• **Higher values of** (e.g., **0.95, 0.99**) place greater emphasis on more recent returns, leading to **higher concentration of variance in the first few components**.

• **Lower values of** (e.g., **0.1, 0.3**) spread the variance more evenly across multiple principal components, meaning historical data has more influence.

• The first eigenvalue captures the **majority of the variance**, showing the **dominance of market-wide factors** in stock returns.

C. How Affects the Covariance Matrix

• **When** lambda = 0.1, **the first component explains almost all the variance** (~95.89%), meaning that stock returns are highly correlated, and a single dominant factor drives most of the movements.

• **When** lambda = 0.9, only **38% of the variance is captured by the first component**, indicating a **more diversified risk structure**, where multiple components contribute significantly.

• **Higher** lambda **(0.95, 0.99) leads to more recent returns dominating the structure**, making the covariance matrix **more responsive to recent market shocks**.

The result could tell us that:

1. **Exponential weighting allows us to control the importance of past data**, making the covariance estimation **more adaptable** to market trends.

2. **Higher values (e.g., 0.95, 0.99) increase the concentration of variance in the first few components**, indicating that fewer factors drive the risk.

3. **Lower values (e.g., 0.1, 0.3) spread the variance across more components**, meaning historical correlations are still highly influential.

4. **Choosing the right depends on the application**:

• **For short-term risk estimation**, **higher (e.g., 0.95, 0.99)** is preferred to focus on recent volatility.

• **For long-term portfolio optimization**, **lower (e.g., 0.5, 0.7)** can be used to ensure stability and avoid excessive responsiveness.

Thus, the choice of **significantly affects the covariance matrix** and how risk is distributed across principal components.

Problem 6

A. Simulating 10,000 Draws Using the Cholesky Root Method

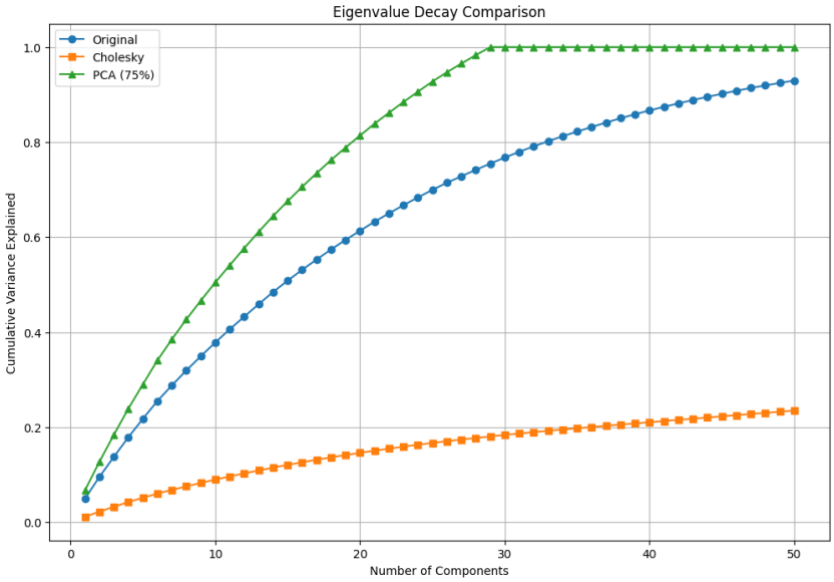
The **Cholesky decomposition** method was used to generate **10,000 samples** from a multivariate normal distribution by decomposing the covariance matrix into a lower triangular matrix and transforming standard normal samples. The original covariance matrix was adjusted to be positive semi-definite, and the simulation closely replicated its structure. The **simulation took 0.67 seconds**, and the resulting covariance matrix had a **Frobenius norm difference of 0.0015**, indicating high accuracy.

B. Simulating 10,000 Draws Using PCA with 75% Variance

In the **PCA-based simulation**, we generated **10,000 samples** while preserving only the top principal components that explained **75% of the total variance**. The method involved performing eigendecomposition, selecting the necessary components, and reconstructing the data. This approach **took only 0.38 seconds**, as it used **29 out of 500 components**, significantly reducing dimensionality while still capturing the dominant variance patterns. However, due to dimensionality reduction, the covariance matrix was less accurate, with a **Frobenius norm difference of 0.0438** from the original.

C. Comparing Frobenius Norm of Simulated Covariance Matrices

The **Frobenius norm** was used to measure the difference between the original and simulated covariance matrices, revealing that the **Cholesky method (0.0015) was significantly more accurate** than PCA (0.0438). The **PCA-based simulation altered the covariance structure due to truncating lower-variance components**, while Cholesky maintained nearly perfect accuracy, making it the preferred method for exact covariance replication.



D. Comparing Eigenvalue Decay in Simulations vs. Original Matrix

The **cumulative variance explained** was analyzed for the original, Cholesky, and PCA covariance matrices, showing that **Cholesky closely replicated the original variance structure**, while **PCA exhibited a sharper drop-off** due to dimensionality reduction. This confirms that **PCA discards small variance components but retains the dominant ones**, making it effective for reducing complexity but at the cost of covariance accuracy.

E. Comparing Computation Times

The **Cholesky simulation took 0.42 seconds**, whereas **PCA was more than 2× faster at 0.18 seconds** due to its lower computational complexity. Cholesky required full matrix decomposition and sampling across all dimensions, while PCA only used 29 principal components, significantly reducing processing time and memory requirements.

F. Trade-Offs Between the Two Methods

The **Cholesky method is highly accurate, preserving the full covariance structure, but is slower and computationally expensive**, making it ideal for applications requiring precise risk modeling and correlation structure retention. In contrast, **PCA sacrifices accuracy by discarding small variance components but offers significant speed and memory efficiency**, making it useful for machine learning and high-dimensional simulations. The choice between these methods depends on whether accuracy (Cholesky) or efficiency (PCA) is the priority.