Forecasting Inflation: A Comparative Study of Econometric and Machine Learning Techniques

Introduction

The objective of this study is to compare various forecasting methods from econometrics and machine learning to address the "curse of dimensionality" presented by macroeconomic datasets with over 100 variables. The analysis begins with a baseline model, the classic AR(1), which serves as a standard for comparison. I then explore whether more complex techniques, such as Lasso, Ridge, VAR, Random Forest, and Principal Component Regression (PCR), can outperform this benchmark. As anticipated, surpassing the AR(1) model proves to be a surprisingly challenging task. However, with careful experimentation and the appropriate application of more advanced methods, it becomes possible to achieve moderate improvements in forecasting performance.

The Data

The dataset I used is FRED_MD, a comprehensive monthly macroeconomic database containing 126 variables, reported from January 1, 1959, to February 1, 2024. These variables are categorized into eight groups: Output and Income, Labor Market, Housing, Consumption, Orders and Inventories, Money and Credit, Interest and Exchange Rates, Prices, and Stock Market. Among these, I selected "All Items Less Food" as the variable of interest, using it as a measure of inflation. Choosing only one variable to forecast greatly simplifies the calculations. The dataset, as of May 2024, is available in my GitHub repository. Alternatively, it can be downloaded directly from the following links:

- FRED-MD and FRED-QD Databases
- St. Louis Fed Research Page

Data Cleaning and Transformation

Upon initial inspection, it was evident that many variables in the dataset suffered from non-stationarity and autocorrelation issues. For instance, the plot of Inflation (CPIULFSL) illustrated these challenges, necessitating transformations to prepare the data for analysis. To address this, I used the fredm function, available through this GitHub repository: FredMD GitHub link. This function, sourced from the FredMD website, efficiently

provided the required adjustments.The effectiveness of these transformations was confirmed by visualizing the stationary Inflation plot and reviewing the corresponding autocorrelation plot.

To further refine the dataset, I addressed missing values and outliers. Variables with extensive missing data were excluded, resulting in the removal of five predictors. Additionally, I excluded the year 1959 and the final observation of 2024 due to significant gaps in the data, ensuring that the analysis began in January 1960. A simple moving average algorithm was applied to impute a small number of scattered missing values within the time series.

Outliers were also a concern, particularly those arising during the COVID-19 period. As the dynamics during this time are considered exogenous to the regular inflationary process, I excluded the relevant observations to avoid their potential influence on the results. These steps ensured that the dataset was clean, stationary, and well-suited for the subsequent forecasting analysis.

Train and Test Samples

I divided the dataset into training and test samples based on a two-thirds split for the training period and the remaining one-third for testing. The training sample covers the period from January 1, 1960, to December 1, 1999, while the test sample spans from January 1, 2000, to December 1, 2019. This approach ensures a sufficient amount of data for model training while reserving an adequate portion for evaluating forecasting performance.

Forecasting Methodology and Approach

The forecasting methodology uses a framework that simulates a real-time forecasting environment. At each iteration, the training dataset is dynamically updated by incorporating the latest observation from the test set. Forecasts are generated using only information available up to the current prediction step, replicating real-world forecasting conditions. The approach is outlined below:

1. Manual Lagging Logic

   Variables are manually shifted to align predictors with the corresponding dependent variable. For the training dataset, the dependent variable (Y_train) is created by removing the first observation, representing future values, while the independent variable (X_train) is created by removing the last observation, representing past values. Similarly, for the test dataset, the dependent variable (Y_test) is formed by

removing the first observation, and the independent variable (X_test) is created by removing the last observation. This ensures temporal alignment in both training and testing.

2. Dynamic Training Data Expansion

   At each iteration, the most recent observation from the test dataset is appended to the training dataset. This dynamic expansion allows the model to continuously adapt to new information as it becomes available.

3. Standardization and De-standardization

   Standardization is performed at each iteration to ensure numerical stability and consistency. The mean and standard deviation are recalculated for the updated training set. Predictions are made on the standardized scale and then transformed back to the original scale for interpretation and evaluation.

4. Model Training and Prediction

   The model is retrained at each iteration using the updated training dataset. For the autoregressive model for example, predictions are explicitly based on the most recent lagged value from the test set. A single prediction is generated for the next observation at each step.

5. Iterative Updates

   After generating a prediction, the actual test observation is compared to the predicted value and appended to the training dataset. This process is repeated until predictions are made for all test observations.

6. Evaluation of Forecast Accuracy

   Forecasts are evaluated against actual test observations using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Residual analyses and visualizations, including time-series plots of actual versus predicted values, provide additional insights into model performance.

This methodology, including the consistent application of manual lagging (by removing the first or last observations as appropriate), ensures a robust and uniform evaluation of different forecasting techniques.

Methods

AR(1)

To analyze the performance of the AR(1) method, I computed two different regressions. In the

first, I used a simple autoregressive model where the observed variable (CPIULFSL) served as the sole predictor. In the second, I expanded the model to include all 121 variables as predictors. When comparing these two approaches, I observed that the first model, relying solely on CPIULFSL, was significantly more precise in forecasting than the model using all variables. This difference was particularly evident in the error metrics, especially the Mean Squared Errors (MSEs), as shown in the table below.

Regression Equation for the AR(1) model:

$$y_t = \alpha + \beta y_{t-1} + \epsilon_t$$

These results highlight the challenges posed by high-dimensional data and underscore the utility of simpler models in forecasting.

Ridge Regression

In Ridge regression, I minimized the sum of squared residuals while adding a penalization term, λ, to reduce the magnitude of the regression coefficients. This penalization effectively controlled model complexity and addressed multicollinearity, which was crucial given the high-dimensional nature of my dataset. The Ridge regression objective function is:

$$\beta = \arg\min_{\beta} \left(\sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right)$$

To evaluate the model, I tested extreme values of λ. When λ is near 0, the penalty has little effect, and the model behaves like ordinary least squares (OLS), minimizing the residual sum of squares. Conversely, as λ approaches infinity, the penalty heavily constrains the coefficients, shrinking them toward 0 and producing a biased, overly simplified model. Drawing on insights from the referenced papers, I identified the optimal λ, which struck a balance between bias and variance. This optimal penalization improved forecast accuracy and demonstrated Ridge regression's ability to combat overfitting in high-dimensional settings.

Lasso

For this project, I examined the behavior of Lasso regression across different levels of penalization, including the Ridge optimal value for comparison. The Lasso regression objective function is given by:

$$\beta = \arg\min_{\beta} \left(\sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right)$$

Ridge and Lasso regression both aim to improve model performance through regularization, behaving like OLS with very small penalties. The key difference is that Ridge shrinks coefficients without eliminating them, whereas Lasso can set some coefficients to exactly zero, performing variable selection.

Principal Component Regression (PCR)

The first step in Principal Component Regression was to evaluate the covariance and correlation matrices of the variables. Visualizations of these matrices, included in the appendix, helped identify the relationships among the variables. To determine the optimal number of components, I computed the eigenvalues and eigenvectors of these matrices. The graph of eigenvalues illustrates the variance explained by each principal component. Based on this analysis, I selected the first five principal components, as the variance contribution from additional components became negligible beyond the sixth. To further evaluate the impact of the number of components, I computed PCR models using 1, 2, 5, and all principal components. The explained variance increased with the number of components, starting at approximately 15% for a single component, 22% for two components, and less than 50% for five components. When all components were used, the MSE matched that of OLS with all predictors, confirming the correctness of the PCA implementation.

Vector Autoregression (VAR)

In this exercise, I employed a Vector Autoregression (VAR) model to capture the interdependencies among multiple time series in the dataset. The VAR model predicts the current value of a variable based on its own lags and the lags of other variables in the system, as described by the following equation:

$$Y_t = \beta_1 Y_t - 1 + \beta_2 Y_t - 2 + \cdots + \beta_p Y_t - p + \varepsilon_t$$

Here, Y_t represents a vector of variables, β_p are coefficient matrices for each lag, and ϵ_t is the error term. By including multiple variables and their interactions, the VAR model is well-suited for capturing the dynamic relationships in macroeconomic datasets. For this analysis, I used lag selection criteria to identify the optimal number of lags, ensuring the model captured meaningful patterns without overfitting. The results highlight VAR's ability to exploit temporal

and cross-variable dependencies, making it particularly effective in forecasting when the relationships among variables are strong.

Random Forest (RF)

Random Forest (RF) was applied as a non-parametric machine learning approach, ideal for handling the high dimensionality and non-linear relationships in the dataset. RF creates an ensemble of decision trees, where each tree produces a prediction, and the final forecast is the average of these predictions:

$$y = \frac{1}{T}\sum_{t=1}^{T} f_t(X)$$

Here, T is the total number of trees, and $f_t(X)$ is the prediction from the each tree. By combining multiple trees, RF reduces the risk of overfitting and improves robustness to noisy data. For this exercise, I fine-tuned hyperparameters such as the number of trees and the maximum depth to optimize performance. The RF model demonstrated its utility in leveraging the large number of predictors and capturing complex non-linear interactions, providing a flexible and effective alternative to traditional econometric models in high-dimensional settings.

Results

Mean Squared Errors		
OLS	With one predictor or AR(1)	1.351572e-05
	With all predictors	1.666667e-05
Ridge	λ = 1e-4	3.889971e-05
	λ = 1e6	1.312739e-05
	λ optimal	1.264374e-05
Lasso	λ = 1e-4	3.873223e-05
	λ = 1e-2	1.173288e-05
	λ = 1e-1	1.198938e-05
	λ = 1	1.312739e-05
	λ = 1e6	1.312739e-05
	λ optimal	1.312739e-05
PCR	With 1 component	1.312878e-05
	With 2 components	1.328557e-05
	With 5 components	1.338589e-05
	With all components	1.666667e-05
RF	1 variable, 500 trees	1.742054e-05
	1 variable, 1 tree	2.33323e-05
	All variables, 500 trees	1.2289e-05
	All variable, 1 tree	2.163237e-05
VAR	All variables	1.28581e-05
	First 30 PC	1.340549e-05

The results highlight the challenge of improving forecasting performance beyond the AR(1) benchmark, given the high dimensionality of the dataset. While more complex methods like Ridge, Lasso, and Random Forest aim to mitigate the curse of dimensionality, their performance depends heavily on the tuning of parameters.

- OLS: The AR(1) model (MSE = 1.351572e-05) outperforms the model using all predictors, underscoring the risk of overfitting in high-dimensional regression.
- Ridge: Ridge regression achieves a lower MSE (1.264374e-05) at the optimal λ, demonstrating its strength in controlling multicollinearity and penalizing overfitting.
- Lasso: Lasso is highly effective at variable selection, achieving the best overall MSE (1.173288e-05) when $\lambda = 0.01$, balancing bias and variance well.
- PCR: The performance of PCR diminishes as more components are included, showing that dimensionality reduction is most effective with a small number of key components.
- Random Forest (RF): RF excels when using all variables with 500 trees (MSE = 1.2289e-05), capturing non-linear relationships better than linear models.
- VAR: VAR shows competitive performance, particularly with all variables (MSE = 1.28581e-05), leveraging cross-variable dynamics effectively.

MSE is one of two metrics used to evaluate forecasting performance, alongside the plot of actual versus predicted values, with visualizations provided in the appendix for the most successful trials. MSE provides a quantitative measure by averaging squared residuals, which places more weight on large errors. As a result, a low MSE might still correspond to poor visual alignment if the model captures stable periods well but fails to track trends or dynamic changes. Conversely, a higher MSE can sometimes reflect good visual alignment if the model captures overall trends but struggles with smaller-scale variations. These differences highlight the importance of using both metrics: MSE offers a statistical summary of error, while the plot provides insights into how well the model captures patterns and variability. Combining these evaluations ensures a more comprehensive understanding of model performance. Lasso and RF stand out as the most effective methods, offering significant improvements over AR(1) by balancing complexity, dimensionality reduction, and predictive accuracy.

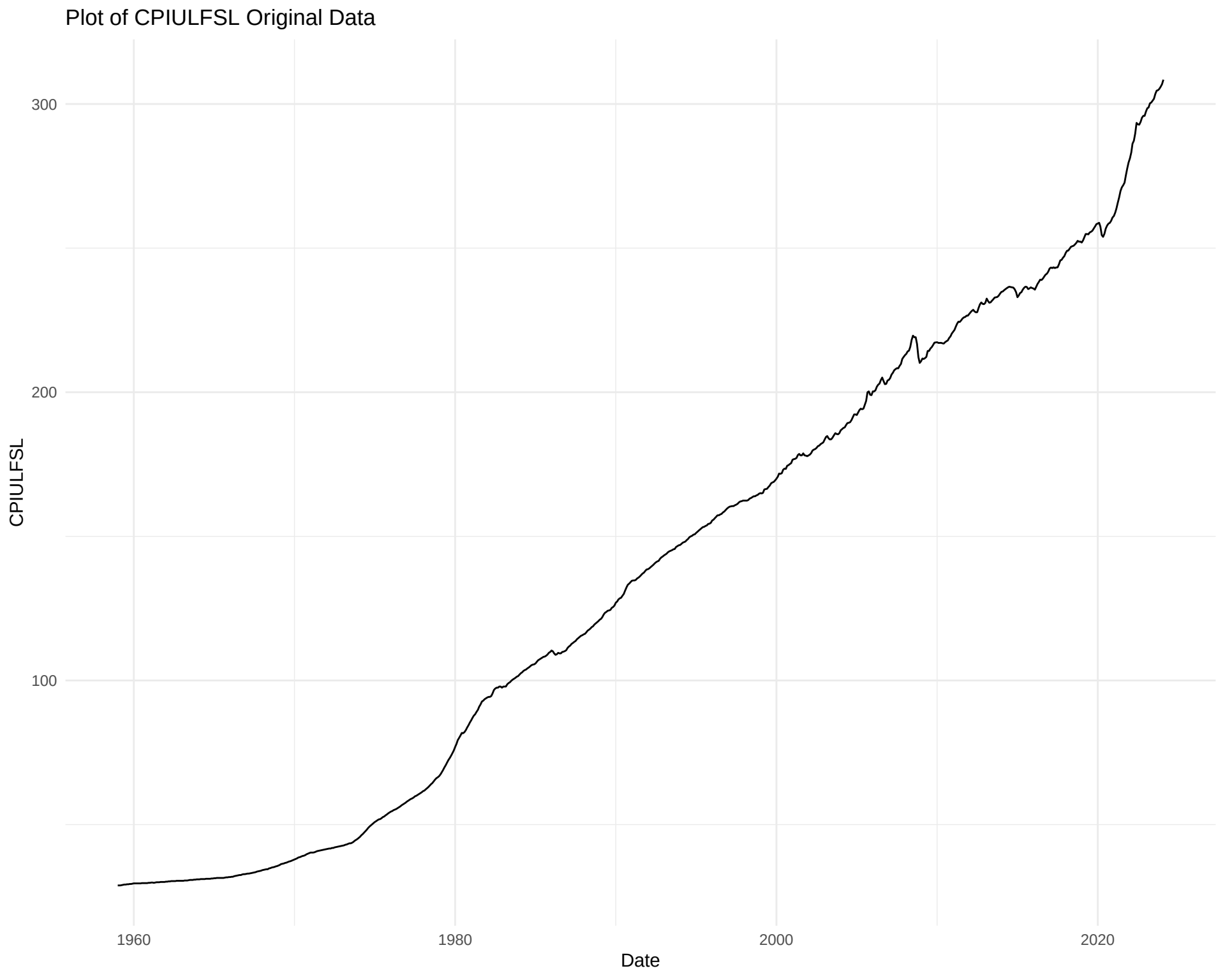
Bibliography

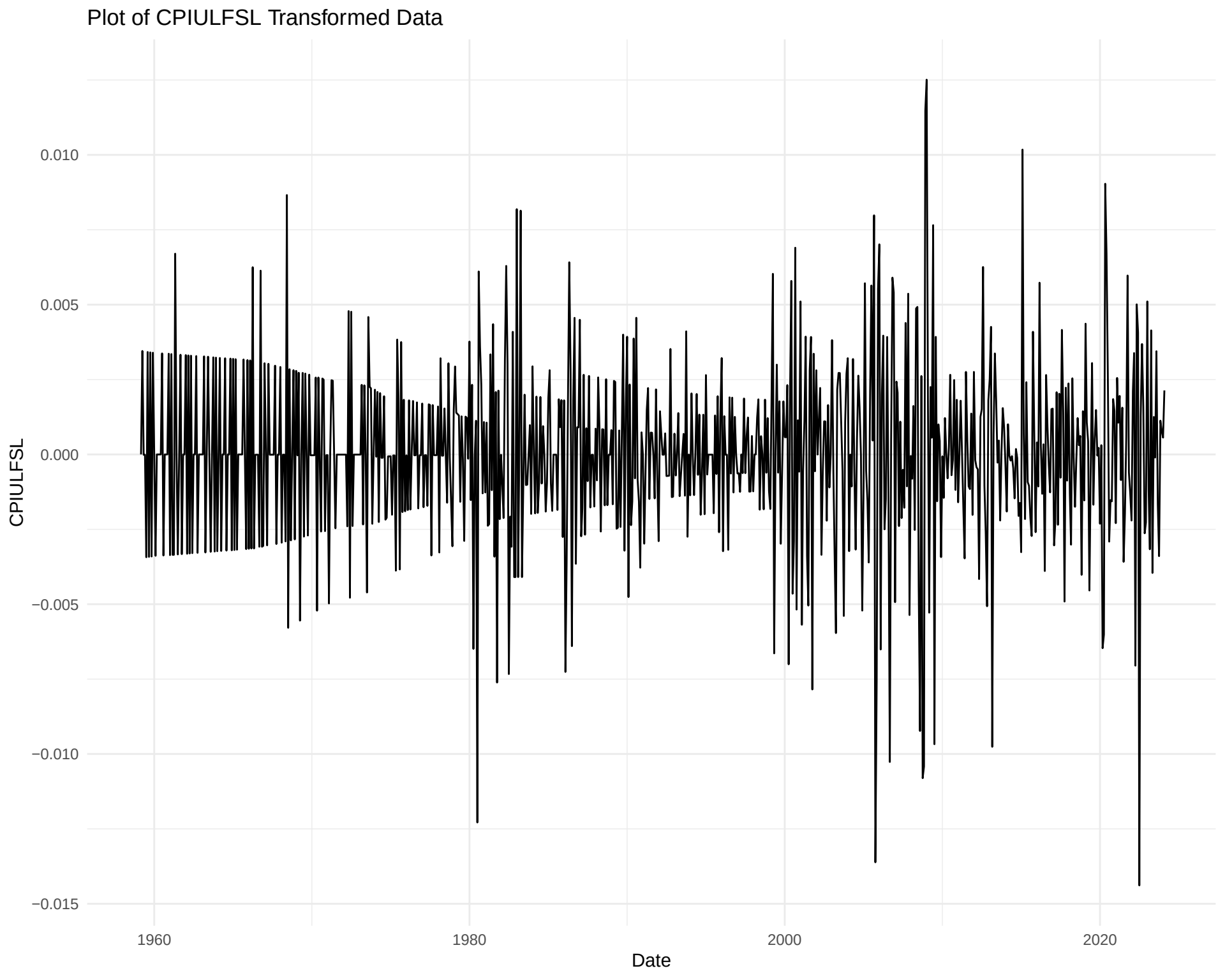
1. **Stock, J. H., & Watson, M. W.** (2020). *Introduction to Econometrics* (4th ed.). Pearson Education.
2. **James, G., Witten, D., Hastie, T., & Tibshirani, R.** (2023). *An Introduction to Statistical Learning with Applications in R* (2nd ed.). Springer.
3. **Mullainathan, S., & Spiess, J.** (2017). Machine Learning: An Applied Econometric Approach. *Journal of Economic Perspectives, 31*(2), 87–106.

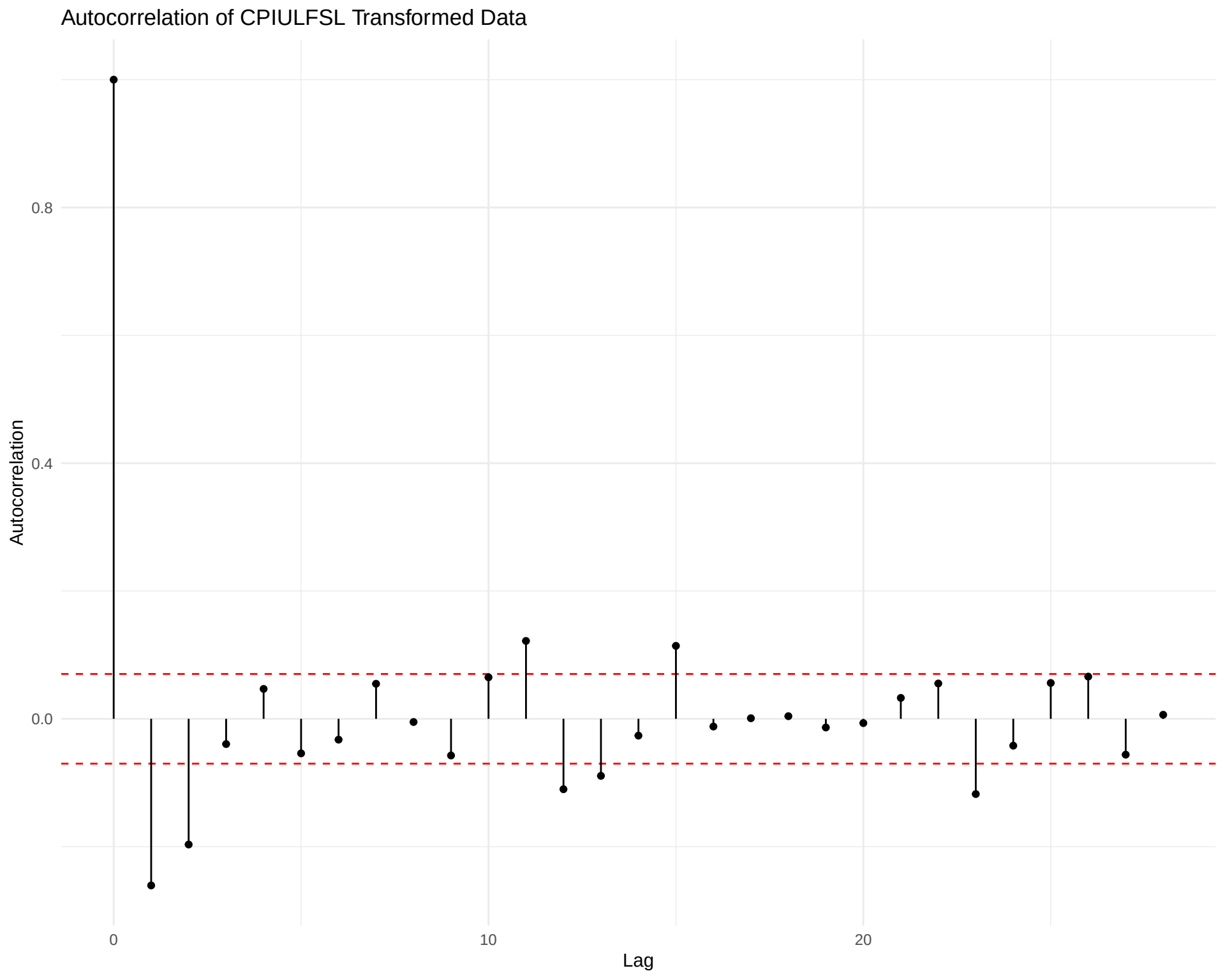
4. **De Mol, C., Giannone, D., & Reichlin, L.** (2008). Forecasting Using a Large Number of Predictors: Is Bayesian Regression a Valid Alternative to Principal Components? *Journal of Econometrics, 146*(2), 318–328.
5. **Stock, J. H., & Watson, M. W.** (2004). An Empirical Comparison of Methods for Forecasting Using Many Predictors. *Unpublished Manuscript.*

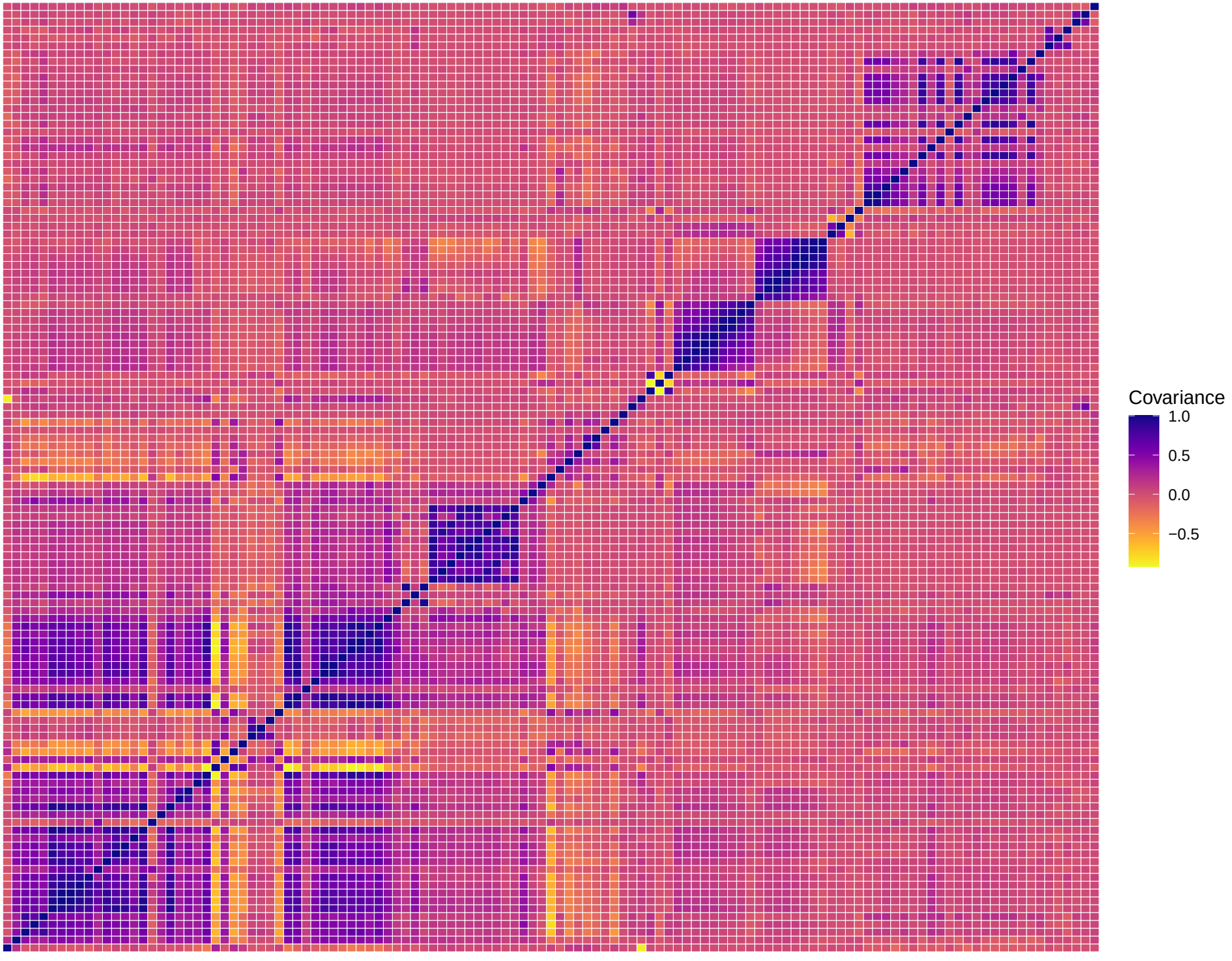
Appendix

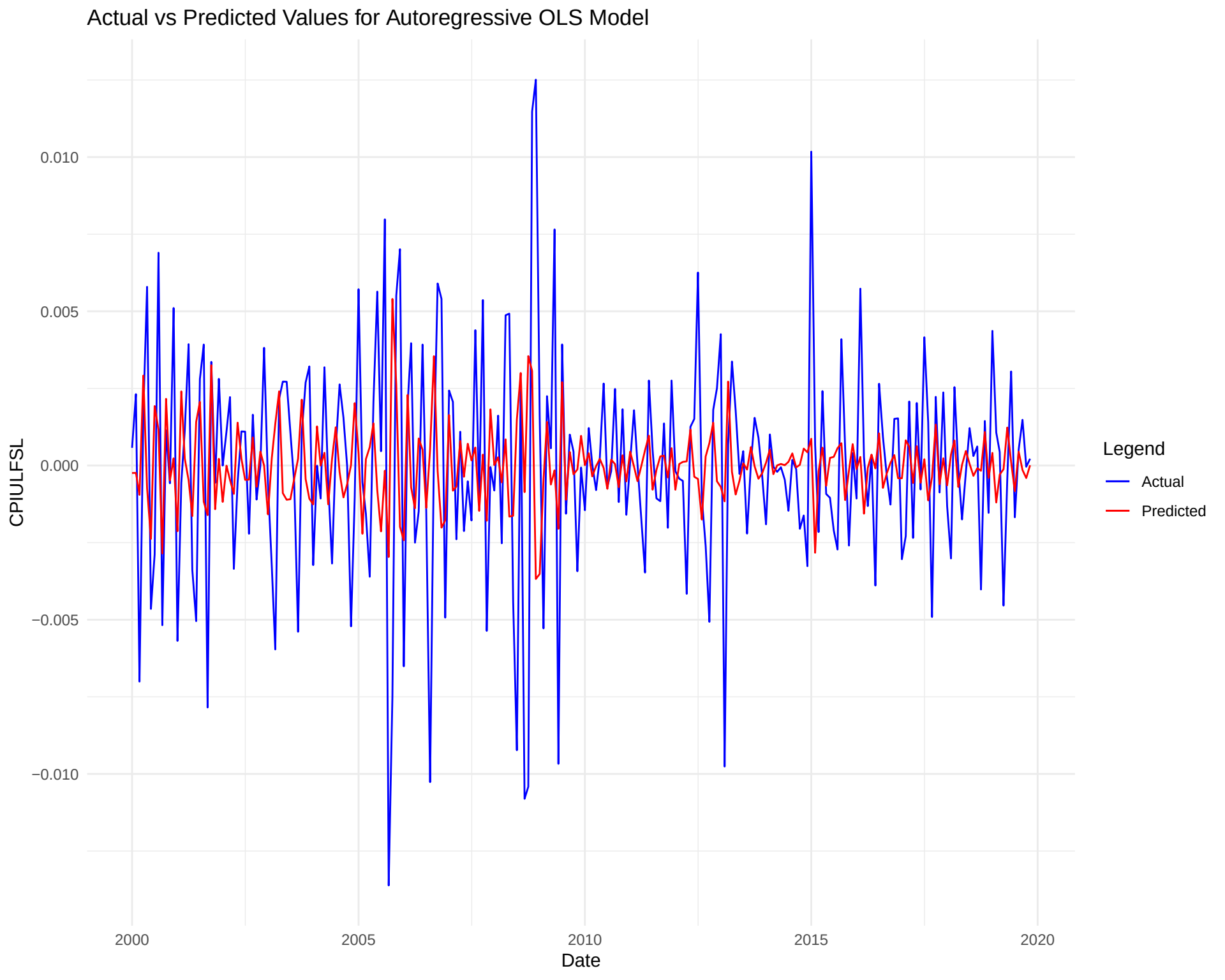
The appendix with the most important plots references in the text will follow at the next page.

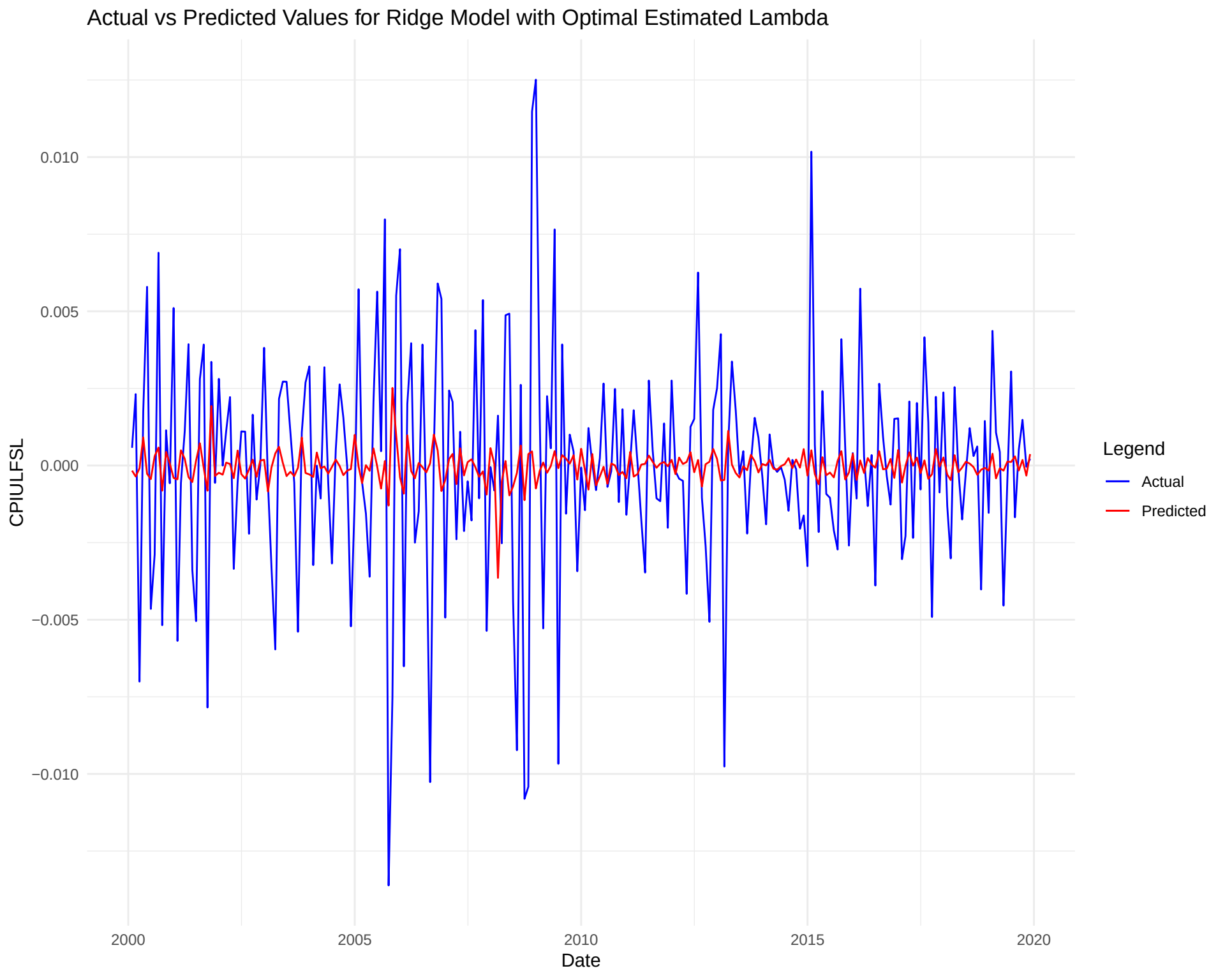

Plot of CPIULFSL Original Data
300
200
100
CPIULFSL
1960
1980
2000
2020
Date

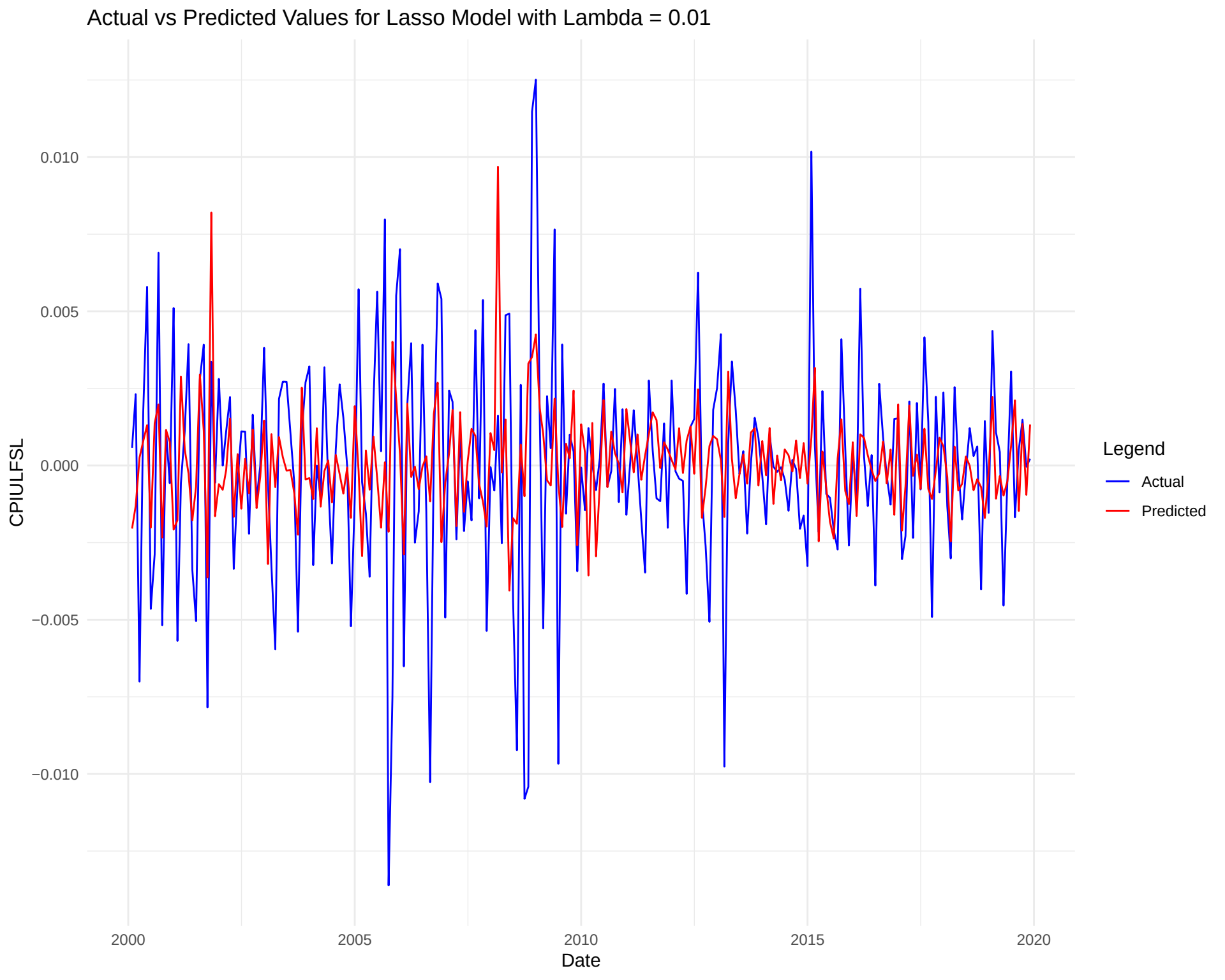

Plot of CPIULFSL Transformed Data
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
−0.015
1960
1980
2000
2020
Date

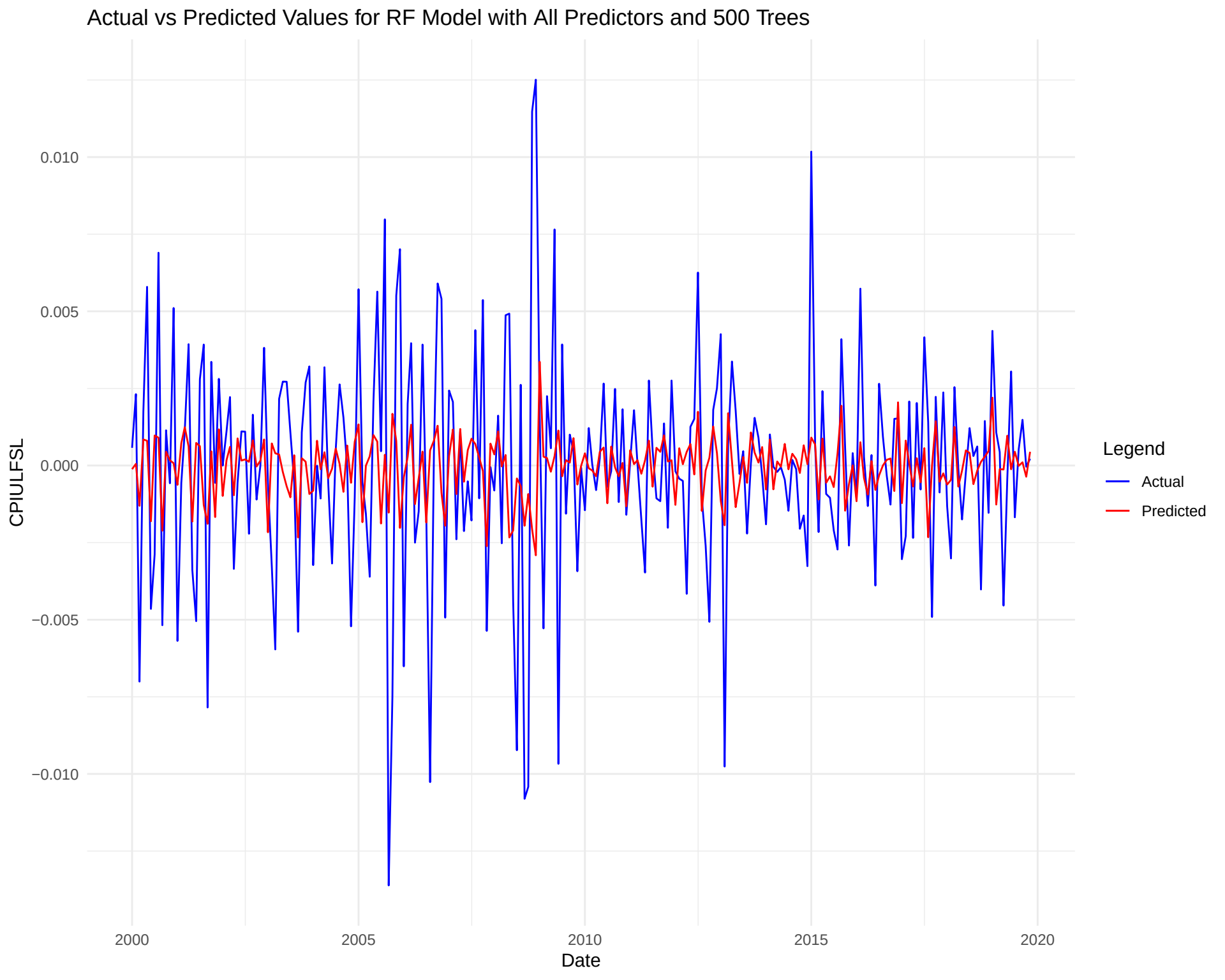

Autocorrelation of CPIULFSL Transformed Data
Autocorrelation
0.8
0.4
0.0
0
10
20
Lag

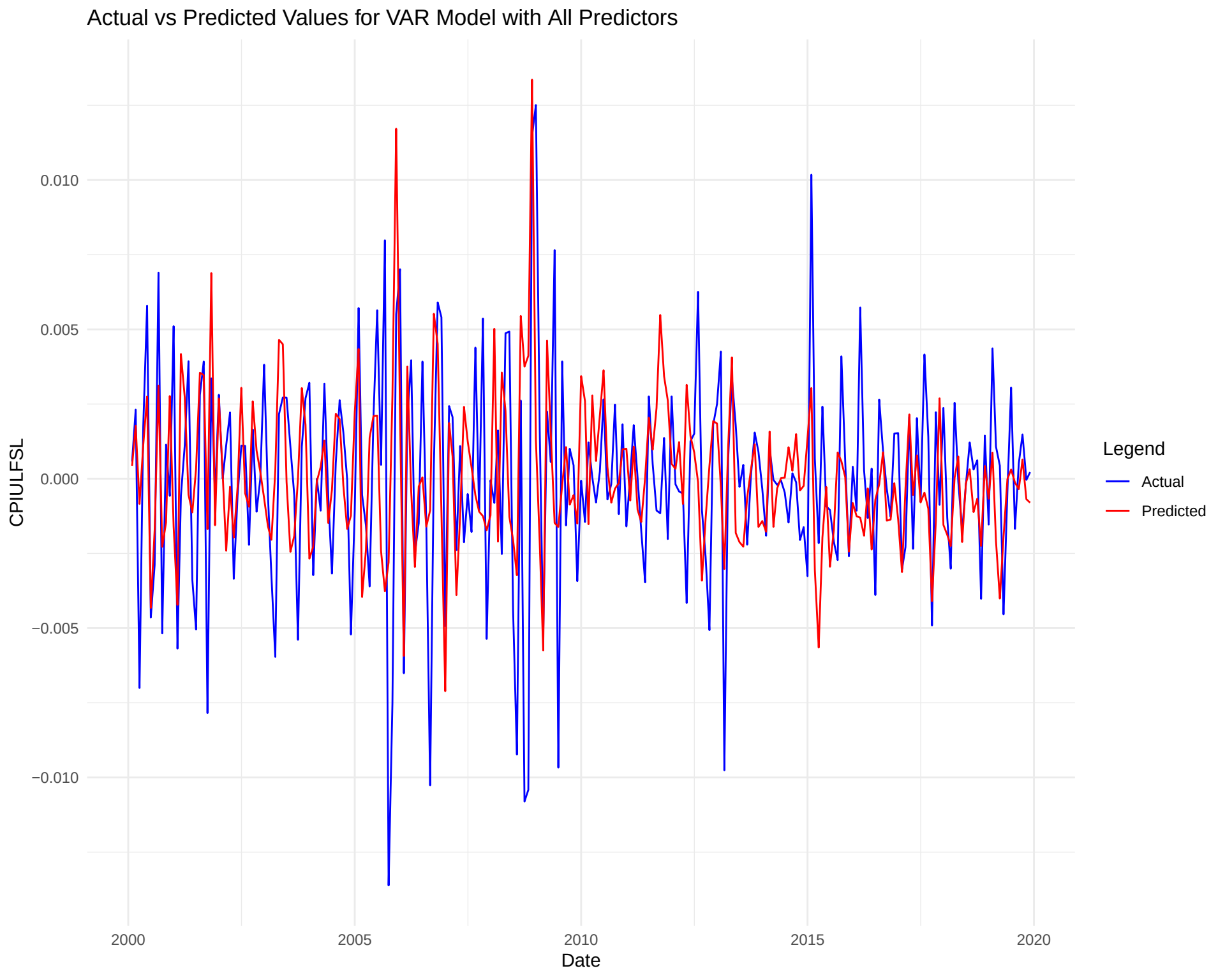

Covariance Matrix Heatmap of FRED_MD Dataset (Standardized)
Covariance
1.0
0.5
0.0
−0.5


Actual vs Predicted Values for Autoregressive OLS Model
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
2000
2005
2010
2015
2020
Date
Legend
Actual
Predicted


Actual vs Predicted Values for Ridge Model with Optimal Estimated Lambda
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
2000
2005
2010
2015
2020
Date
Legend
Actual
Predicted


Actual vs Predicted Values for Lasso Model with Lambda = 0.01
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
2000
2005
2010
2015
2020
Date
Legend
Actual
Predicted


Actual vs Predicted Values for RF Model with All Predictors and 500 Trees
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
2000
2005
2010
2015
2020
Date
Legend
Actual
Predicted


Actual vs Predicted Values for VAR Model with All Predictors
CPIULFSL
0.010
0.005
0.000
−0.005
−0.010
2000
2005
2010
2015
2020
Date
Legend
Actual
Predicted