Time Series Analysis: Forecasting the Consumer Price Index for Urban Consumers in the U.S.

PSTAT 274 Final Project

Taryn Li

Instructor: Raya Feldman

June 9, 2023

Abstract

The report aims to analyze the monthly time series dataset of the Consumer Price Index for all urban consumers (CPI-U) ranging from May 2009 to April 2023 that was collected by the U.S. Bureau of Labor Statistics (BLS). The CPI-U is an important index in financial economics. It serves for tracking inflation, understanding the cost of living, shaping economic policies, and facilitating informed decision-making for individuals, businesses, and even governments, so forecasting the CPI-U brings much convenience for people in the financial and economic markets.

In the report, I analyze the time series first and make some modifications such as transformation and differencing to improve the data for better setting up the models. By applying autocorrelation functions (ACFs), partial autocorrelation functions (PACFs), AICc (Akaike Information Criterion, corrected for bias), multiple useful tests, and other analysis (i.e. spectral analysis), I eventually choose a model ARIMA(2,1,4) that passes almost all diagnostic tests for predicting the CPI-U. My results of predicting the last 12 observations (12 months) of the whole data show that the predictions are within the bounds and verify that this model is an appropriate model for forecasting CPI-U.

1 Introduction

The Consumer Price Index for all urban consumers is a widely used measure of inflation in the United States that measures the average price fluctuations over time for a basket of goods and services purchased by urban households. CPI-U is so crucial in the financial and economic market. It contributes to economic

planning, informed decision-making, risk management, investment strategies, and government policy formulation. To be specific, it can help stakeholders anticipate future price movements and take proactive measures to navigate the impact of inflation on various aspects of the economy.

The Consumer Price Index for urban consumers data is found on FRED Economic Data and is collected by the U.S. Bureau of Labor. The raw data monthly records the CPI-U from January 1947 to April 2023. The report focuses on the recent 14 years of monthly data. In other words, the original time series data in this report is from May 2009 to April 2023.

By using R studio, I filter the raw data to the target range of the data and regard it as the original data throughout the whole report. On R studio, I convert the original data to the time series and plot it. The data is indeed a time series because the observations are recorded sequentially over time. Then, I split the time series into the training and testing time series data. The testing data is the last 12 observations of the original data (monthly data from May 2022 to April 2023). Most analysis of the report is based on the training time series. I analyze the original training time series by showing its characteristics on the histograms and plots. I find that the series is not stationary and has an apparent trend. To fix these issues, I use Box-Cox transformation to stabilize the variance and one differencing at lag 1 to remove the trend. After, I get detrended and stationary training data.

Using the detrended and transformed data, I plot the ACF and PACF, which is a vital step to start the journey to find the candidate models for predictions. AICc is a good criterion for me to select 6 possible models with the lowest AICcs. Next, I use Maximum Likelihood Estimation Method (MLE) to fit the models and derive parameter estimations. I also use various tests and checking. Finally, I get a best-fit model, which is ARIMA(2,1,4), to forecast the future 12 observations (or the last 12 observations). The plots show that the predictions given by the model are well fit in the bounds, which is a nice result.

2 Exploratory Data Analysis

There are 168 observations in the dataset ranging from 2009 to 2023. Using the recent 14-year data can better help achieve the goal because it gives more relevance to current market conditions and a better representation of relevant factors. To begin with, I split the data into training and testing sets because this step is crucial for assessing the performance of models accurately. From Figure 2.1, the training time series shows an obvious upward trend. The red line indicates the clear tendency of the trend. As time goes on, the monthly price index goes up. Also, the variance and mean are not constant. Typically, the variance fluctuates over time. There is no strong seasonality in the data. There are also no sharp changes.

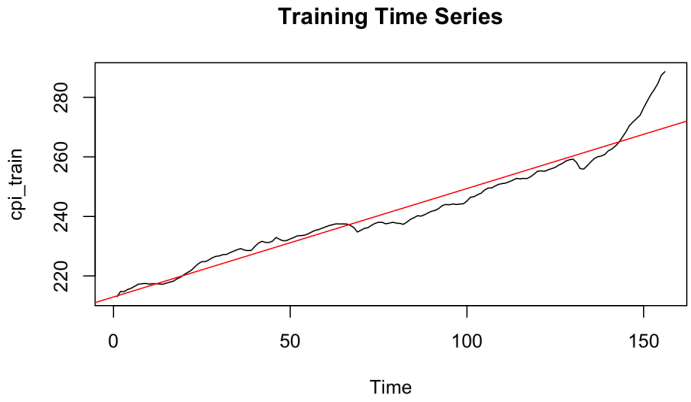

Figure 2.1: Plot of Training Time Series Data

The decomposition plot of the training data shown in Figure 2.2 exhibits more characteristics. A strong trend pattern undoubtedly exists.

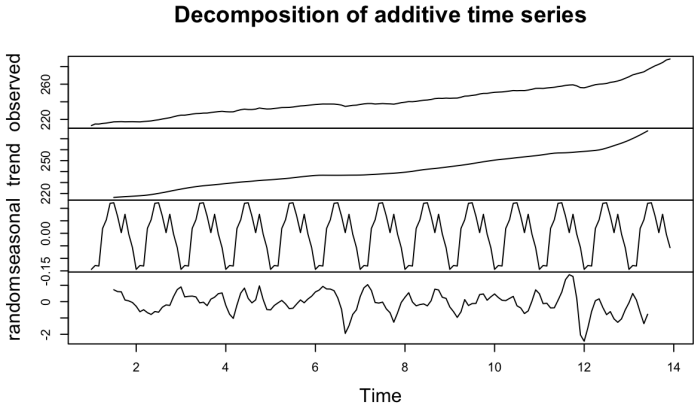

Figure 2.2: Decomposition Plot of Training Time Series

Above all, I can conclude that the training time series has a nonconstant mean, unstable variance, and a strong upward trend. The training data is highly nonstationary as well.

3 Transformation and Differencing

The purpose of this section is to work out the existing issues of the training time series: highly non-stationary problem (nonconstant variance) and the trend pattern.

3.1 Variance Stabilization

Suppose the original training time series data be X_t. Since the training data is not stationary and the distribution is skewed, Box-Cox transformation can be used. 'bcTransform' command gives the suggested value of lambda, which is $\lambda = -2$. Through Figure 3.1, I fail to find any interpretable value of λ, I choose to use the Box-Cox transform with $\lambda = -2$, which gives the new time series

$$Y_t = \frac{1}{\lambda}(X_t^{\lambda} - 1) = \frac{1}{-2}(X_t^{-2} - 1).$$

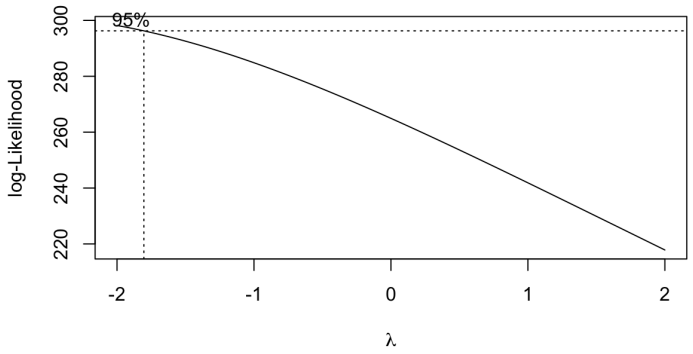

Figure 3.1: Box-Cox Transform λ

The plot of the transformed data shows a more stable variance than the plot of the training data does. Also, the Box-Cox transform gives a more symmetric, normal histogram and more even variance. These are shown in Figure 3.2. So far, the variance has been stabilized.

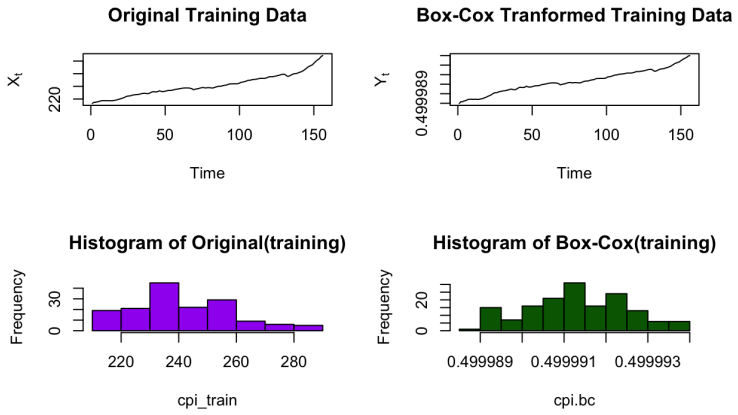

Figure 3.2: Plots & Histogram of 2 Training Data (original vs. box-cox)

3.2 Trend Elimination

In addition to the unstable variance issue, there is a trend in the training data, so I use differencing at lag 1 to eliminate the trend. Using the first difference of the training set at lag 1 is effective because the sample variance decreases from 1.379884e-12 to 1.73068e-15. As I employ the second difference, there is a rise in the sample variance from 1.73068e-15 to 1.734137e-15, revealing that the second difference of the training set is unnecessary. The change is small, but it accounts for an increase.

I perform the Box-Cox transformation with $\lambda = -2$ to make the variance constant and use the first difference at lag 1 to remove the trend. From Figure 3.3, the training time series is relatively stationary now. There is no trend and no seasonality in the improved training data (after transformation and one differencing).

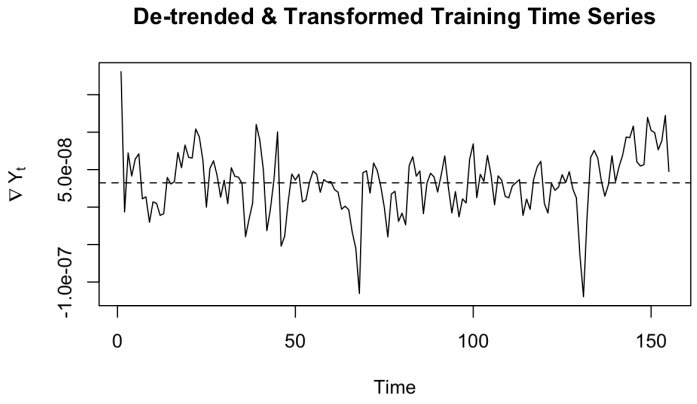

Figure 3.3: Plot of De-trended Transformed Training Time Series

Furthermore, the histogram of the improved training data reveals that it follows a Gaussian and almost symmetric distribution (shown in Figure 3.4).

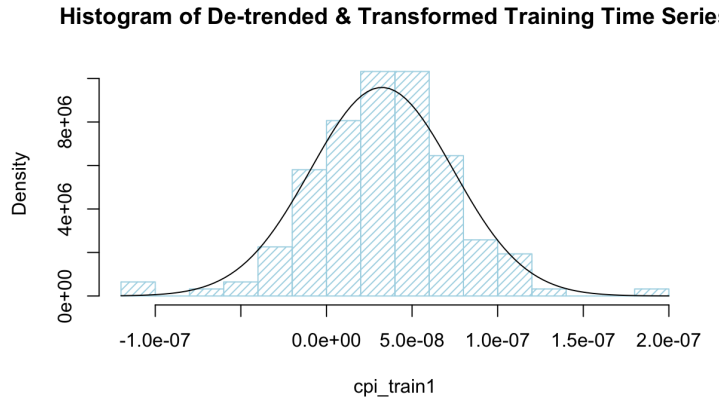

Figure 3.4: Histogram of De-trended Transformed Training Time Series

4 Model Identification & Estimation

In this section, some candidate models are determined. I select possible models and find the estimation for those models.

4.1 Model Identification

It is time to identify the models I need to use to predict data. There is no seasonal part in this training data, so the possible models I can consider are MA(q) (Moving Average Models), AR(p) (Autoregressive Model), ARMA(p,q), and ARIMA(p,d,q). p, d, q are the parameters that can determine the specific models I need in the following process. Since the original training data is not stationary and has a trend, I apply the first difference to remove the trend, so d = 1. Then, to estimate the parameters of p and q, it is significant to use autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs). From Figure 4.1, the ACF shows statistical significance at lag 1, 2, and 5. The PACF shows statistical significance at lag 1 and 4. Then, I calculate AICcs for ARIMA (Autoregressive Integrated Moving Average) models (with d = 1) with p running from 0 to 5 (based on the PACF) and q running from 0 to 6 (based on the ACF). I identify 6 possible models with the lowest AICcs:

$$\text{ARIMA(5,1,4) (AICc = -4870.295)}$$
$$\text{ARIMA(5,1,5) (AICc = -4868.511)}$$
$$\text{ARIMA(2,1,4) (AICc = -4867.510)}$$
$$\text{ARIMA(2,1,3) (AICc = -4867.001)}$$
$$\text{ARIMA(5,1,6) (AICc = -4866.545)}$$
$$\text{ARIMA(3,1,4) (AICc = -4866.479)}$$

Followed by the above sequence, I identify them as Model 1, Model 2, Model 3, Model 4, Model 5, and Model 6.

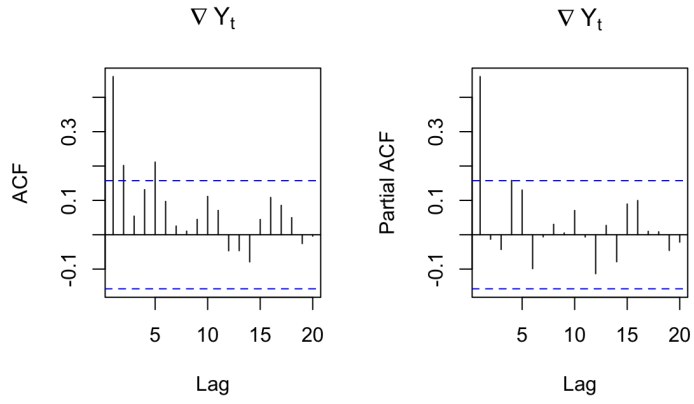

Figure 4.1: ACF and PACF of De-trended Transformed Training Time Series

4.2 Model Estimation

I fit these six models using Maximum Likelihood Estimation (MLE/ML). I also calculate the 95% confidence intervals

$$|\hat{\rho}(k)| < |\frac{1.96}{\sqrt{n}}| = |\frac{1.96}{\sqrt{156}}| \approx 0.1569256$$

to assess the coefficients of the models. If the coefficient estimated by MLE is within the 95% confidence interval, that coefficient could be 0. Then fixing the model with that 0 coefficient can create a new model. Comparing the AICs/AICcs of the original model and the new model can help choose which of these two models is better. The smaller the AIC/AICc is, the better the model is.

ARIMA(5,1,5) and ARIMA(5,1,6) are considered bad models because they fail to pass Box-Pierce, Ljung-Box, and the Shapiro-Wilk normality test. ARIMA(5,1,4) and ARIMA(2,1,3) are also excluded because their ar1 equals 0. In this case, d = 1 means that in the AR part, I have $1 - B$. If ar1 equals 0, the coefficient of ar1 will then become -1, leading to failure to pass the unit roots tests. The root is on the unit circle, so the model is not stationary. Stationarity is crucial for a good forecasting model. Therefore, only ARIMA(2,1,4) and ARIMA(3,1,4) can be considered in the further steps.

ARIMA(2,1,4) (Model 3) is special because ma1, ma2, and ma4 could be 0. This means that there is a new model for it. Figure 4.2 clearly states that the AIC of the new model is smaller than that of the original model, so I choose the new model and I regard it as Model 3 new.

```
Call:
arima(x = cpi_train1, order = c(2, 1, 4), method = "ML")

Coefficients:
Warning: NaNs produced          ar1      ar2      ma1    ma2      ma3     ma4
      -0.3107  -0.4027  -0.1498  0.037  -0.5485  -0.123
s.e.      NaN   0.2265      NaN    NaN      NaN     NaN

sigma^2 estimated as 1.283e-15:  log likelihood = 2421.07,  aic = -4830.14

Call:
arima(x = cpi_train1, order = c(2, 1, 4), fixed = c(NA, NA, 0, 0, NA, 0), method = "ML")

Coefficients:
          ar1      ar2  ma1  ma2      ma3  ma4
      -0.5046  -0.4130    0    0  -0.6130    0
s.e.   0.0904   0.0989    0    0   0.0938    0

sigma^2 estimated as 1.312e-15:  log likelihood = 2419.44,  aic = -4832.88
[1] -4830.309
```

Figure 4.2: Estimations of Model 3 and Model 3 new

So far, the chosen fit models:

Model 3 new: ARIMA(2,1,4) (AICc = -4830.309)

Model 6: ARIMA(3,1,4) (AICc = -4833.556)

In fact, both Model 3 new and Model 6 pass Box-Pierce, Ljung-Box, and McLeod-Li tests, but they fail to pass the Shapiro-Wilk normality test because the p-values are less than 0.05. I reject the null hypothesis. Thus, the residuals are not normally distributed. This can be further considered in future learning. Based on what I have learned, Model 3 new and Model 6 are candidate models that need to be diagnostics checked again.

5 Model Diagnostics

Diagnostics checking is one of the important steps in finding the candidate models. I need to confirm the 2 main properties of the models. One is the stationarity and invertibility of the models. They ensure the reliability and accuracy of the forecasts. For stationarity, some models like ARIMA require the data to be stationary. Violation of stationarity assumptions can lead to biased forecasts and unreliable model performance. For invertibility, it ensures that the past values of the forecast errors can be used to estimate the current value. Hence, it is essential to check stationarity and invertibility. The other one is checking residuals. Residuals are the differences between the observed values and the values predicted by the forecasting model. By examining the residuals, we can check the assumptions of the models, including white noise assumptions, normality of residuals, independence of errors, and absence of autocorrelation. Now, based on these two key points, I can diagnose the two chosen models in the last section. Note: $Y_t = \frac{1}{-2}(X_t^{-2} - 1)$.

5.1 Model 3 new

Algebraic form for fixed ARIMA(2,1,4) (AICc = -4830.309):

$$(1 + 0.5046B + 0.4130B^2)(1 - B)Y_t = (1 - 0.6130B^3)Z_t$$

First, I check the stationarity and invertibility of Model 3 new by using unit roots tests. From Figures 5.1 and 5.2, I can observe that all roots are outside the unit circle in both MA and AR parts. That means that the new Model 3 is stationary and invertible. This model passes the unit roots test.

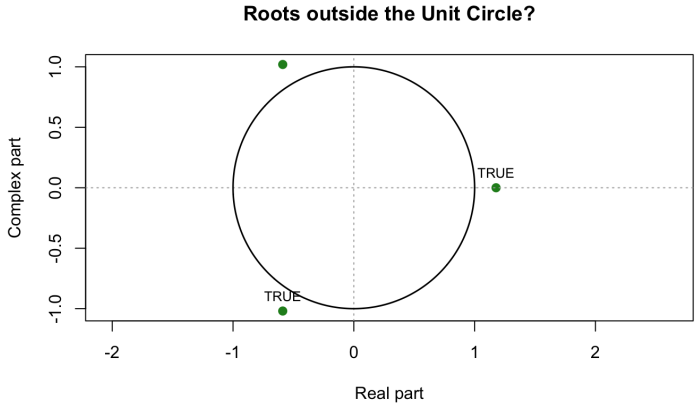

Figure 5.1: Unit Roots in MA Part for fixed ARIMA(2,1,4)/Model 3 new

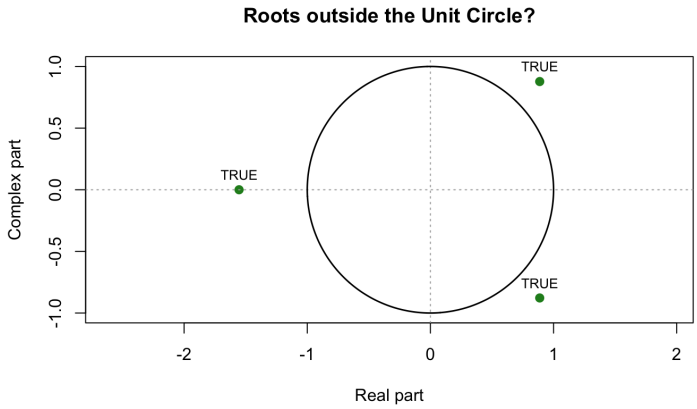

Figure 5.2: Unit Roots in AR part for fixed ARIMA(2,1,4)/Model 3 new

Then, I analyze the residuals by graphing the plots. Figure 5.3 shows that all acfs and pacfs of the residuals for the new Model 3 are within the confidence intervals. q-q plot is nearly a straight line. There is no trend, no seasonality, and no change in variance of Model 3 new. Thus, the residuals appear to be white noise. Furthermore, by Box-Pierce, Box-Ljung, and McLeod-Li tests, I can check the independence of residuals. The results of these tests all point out that p-values are greater than 0.05, so residuals are independent. However, by the Shapiro-Wilk normality test, the p-value is less than 0.05, which means that the residuals are not normally distributed. It is possible to have such a situation. At least, what I have learned from 274 cannot solve this issue further.

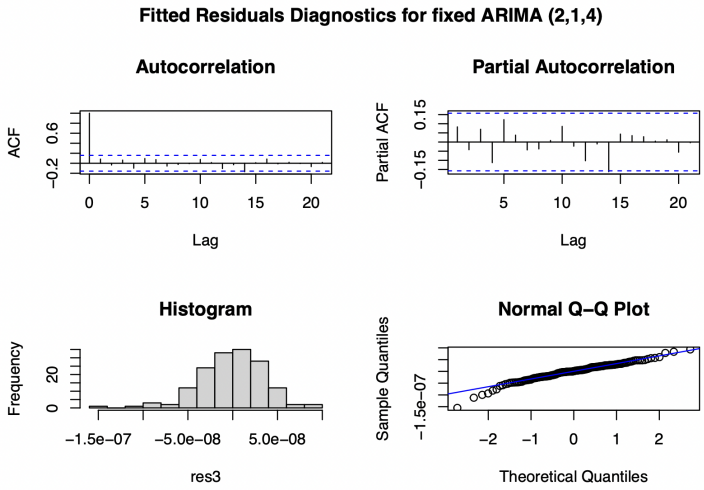

Figure 5.3: Fitted Residuals Diagnostics for Model 3 new

5.2 Model 6

The diagnostic checking process of Model 6 is the same as that of Model 3 new.

Algebraic form for ARIMA(3,1,4) (AICc = -4833.556):

$$(1 + 0.1244B + 0.3881B^2)(1 - B)Y_t = (1 - 0.5257B - 0.6262B^3 - 0.0668B^4)Z_t$$

By applying the unit roots test, Figures 5.4 and 5.5 reveal that Model 6 fails to pass the test because in each part (MA and AR), there is one root inside the unit circles. Thus, Model 6 is not stationary and invertible. Model 6 now can be excluded because it is not a good forecasting model in this case. It is unnecessary to do further residual analysis.

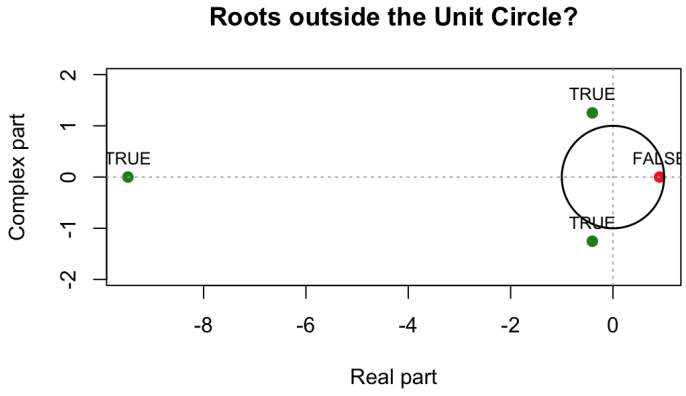

Figure 5.4: Unit Roots in MA part for ARIMA(3,1,4)/Model 6

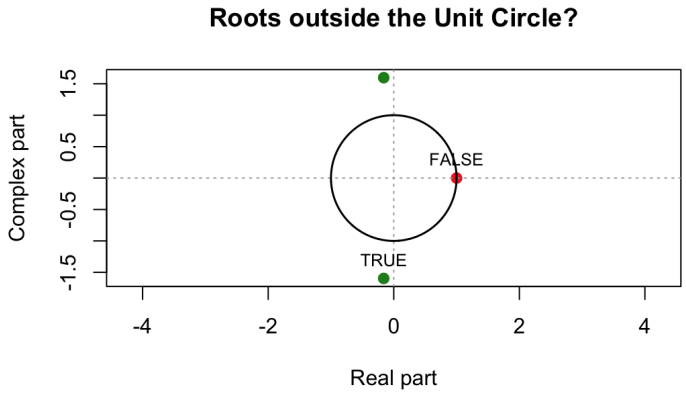

Figure 5.5: Unit Roots in AR part for ARIMA(3,1,4)/Model 6

6 Spectral Analysis

The spectral analysis includes three parts: periodogram, Fisher's test, and Kolmogorov-Smirnov test.

6.1 Periodogram

Periodogram is a valuable tool for detecting the presence of seasonal patterns within a time series by identifying recurring periods, but the data in this report fails to show strong seasonality. However, I can still use periodogram to further

check this condition. From Figure 6.1, the periodogram of residuals shows no dominated frequency. In a sense, I double check there is no strong seasonality in this data.

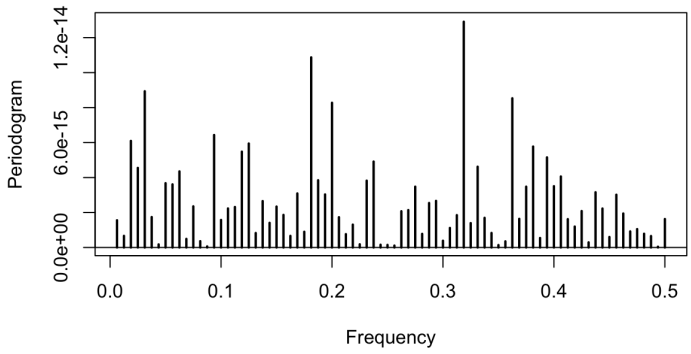

Figure 6.1: Plot of Periodogram for Residuals for new ARIMA(2,1,4)

6.2 Fisher's Test & Kolmogorov-Smirnov Test

The Fisher and Kolmogorov-Smirnov tests are utilized to ascertain whether a time series exhibits attributes of white noise. If the residuals resemble white noise, which is typically observed in a well-fitted model, they are expected to successfully pass both tests. For the Fisher's test, since the p-value = 0.4881152, which is greater than 0.05, it passes the test. The residuals for the chosen fit model resemble Gaussian white noise. For the Kolmogorov-Smirnov test, Figure 6.2 shows that it passes the test as well.

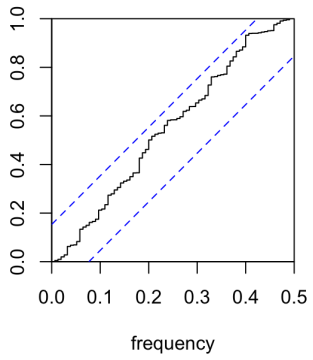

Figure 6.2: Plot Derived by the Kolmogorov-Smirnov Test

7 Forecasting

After doing so many steps, I obtain an optimal forecasting model based on the training data. Now, I use Model 3 new/ fixed ARIMA(2,1,4) to predict the monthly CPI-U, or we say the consumer price index for urban people, from May 2022 to April 2023. Figure 7.1 shows the predictions based on the transformed training data. The red points are the predictions. The blue dash lines represent the confidence intervals. I find that our predictions are within the intervals, which indicates that this is a relatively good forecast. Similarly, Figure 7.2, which shows the predictions based on the original training time series, has the same result as that in Figure 7.1. Figure 7.3 is a zoomed version of Figure 7.2. The difference is that I add green points. These green points are the testing data. In other words, they represent the real values. As I observe, there are still some deviations between the predicted values and the real values, but they are already relatively close. Perhaps fixed ARIMA(2,1,4) /Model 3 new is the best model among all the models I have learned. In the future, I hope to learn more knowledge and models to make the difference between the predicted and true values even smaller and tend to be zero. Above all, this model works as a good forecasting model to predict the CPI-U.

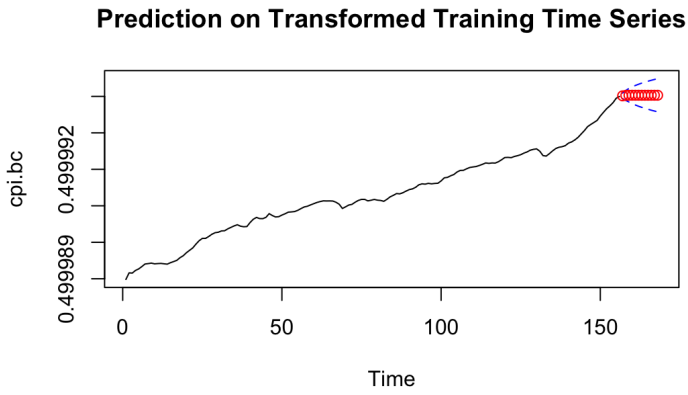

Figure 7.1: Graph of Predictions on Transformed Training Data

Figure 7.2: Graph of Predictions on Original Training Data

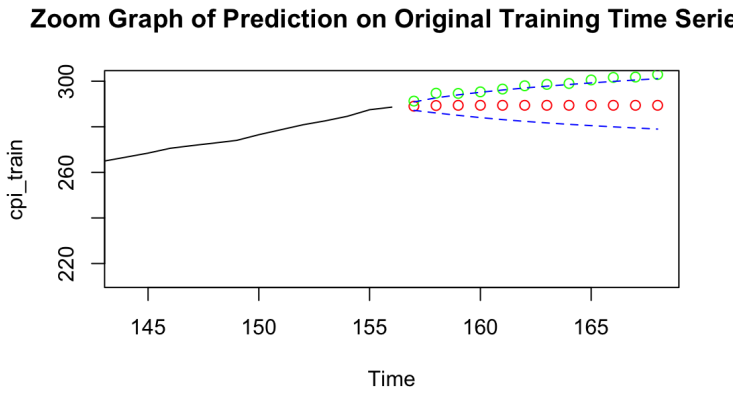

Figure 7.3: Zoomed Graph of Predictions on Original Training Data with Real Values (Testing Data)

8 Conclusion

In conclusion, this report analyzes the monthly time series dataset of the Consumer Price Index for all urban consumers (CPI-U) collected by the U.S. Bureau of Labor Statistics (BLS) from May 2009 to April 2023. The goal is to forecast the CPI-U, which is a crucial index in financial economics that helps track inflation, understand the cost of living, shape economic policies, and facilitate informed decision-making.

The report starts with an exploratory data analysis, where the training time series data is analyzed for its characteristics. It is observed that the data has a clear upward trend, non-constant mean and variance, and is non-stationary. To address these issues, the data undergoes transformation and differencing. Box-Cox transformation is used to stabilize the variance, and one differencing is applied to remove the trend. The transformed and detrended training data shows stability and stationarity.

Next, model identification and estimation are performed. Autocorrelation functions and partial autocorrelation functions are used to identify possible models. AICc is used as a criterion to select the best models, resulting in two candidate models: fixed ARIMA(2,1,4) and ARIMA(3,1,4) (Model 3 new and Model 6). Maximum Likelihood Estimation (MLE) is used to estimate the parameters of these models.

Model diagnostics are then conducted to assess the chosen models. Stationarity and invertibility tests are performed, confirming that the models are

suitable. The residuals of the models are examined for white noise assumptions, normality, independence, and absence of autocorrelation. While the residuals appear to be white noise, independent, and free from trend and seasonality, they do not follow a Gaussian distribution.

Based on the diagnostic checks, the fixed ARIMA(2,1,4) model (Model 3 new) is deemed the most appropriate for forecasting the CPI-U. This model passes most diagnostic tests and demonstrates accurate predictions for the last 12 observations of the original data. However, it is worth noting that the residuals of this model do not meet the normality assumption.

In summary, the analysis and forecasting of the CPI-U time series dataset provide valuable insights into inflation and economic trends. The fixed ARIMA(2,1,4) model proves to be a suitable choice for predicting future CPI-U values. However, further research could focus on improving the normality of residuals to enhance the model's performance. The difference between the predicted values and true values may be caused by the lack of normality in the model. Overall, this report contributes to forecasting the CPI-U, resulting in informed decision-making as well as a more comprehensive understanding of the impact of inflation on various aspects of the economy.

In the end, I want to thank Professor Raya Feldman, TAs Lihao Xiao, and Thiha Aung for their help and great suggestions on my final project.

9 References

U.S. Bureau of Labor Statistics, Consumer Price Index for All Urban Consumers: All Items in U.S. City Average [CPIAUCSL], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/CPIAUCSL, June 8, 2023.

Feldman, Raya. PSTAT 174/274 Lecture Slides Week 1 - 9, Spring 2023.

10 Appendix

```r
# install.packages("devtools", repos = "http://cran.us.r-project.org")
# devtools::install_github("FinYang/tsdl")
# install.packages("tidyverse")
# install.packages("dplyr")
# install.packages("lubridate")
# install.packages("ggplot2")
# install.packages("ggfortify")
# install.packages("qpcR")
# install.packages("UnitCircle")
# install.packages("TSA")
# install.packages("GeneCycle")
library(tsdl)
library(tidyverse)
library(dplyr)
library(MASS)
library(lubridate)
library(forecast)
library(ggplot2)
library(ggfortify)
library(qpcR)
library(UnitCircle)
library(GeneCycle)
```

```r
# Load and filter raw dataset
cpi <- read.csv("~/Desktop/PSTAT 274/Final Project/data/CPIAUCSL.csv", sep=",")
cpi.csv <- cpi[as.Date(cpi$DATE) >= as.Date("2009-05-01")
                    & as.Date(cpi$DATE) <= as.Date("2023-04-01"), ]
head(cpi.csv)

write.csv(cpi.csv, file = "~/Desktop/PSTAT 274/Final Project/data/cpi.csv",
          row.names = FALSE)

nrow(cpi.csv)  # number of observations in the original time series
cpi.ts <- ts(cpi.csv[,2], start = c(2009,1), frequency = 12)

# Split the original data
cpi_train <- cpi.ts[c(1: 156)]
cpi_test <- cpi.ts[c(157: 168)]

# Time series plot of training set
ts.plot(cpi_train, main = "Training Time Series")
fit <- lm(cpi_train ~ as.numeric(1:length(cpi_train)))
```

```r
abline(fit, col = "red")

# Decomposition plot of training time series
y <- ts(as.ts(cpi_train), frequency = 12)
decomp <- decompose(y)
plot(decomp)

hist(cpi.ts, main = "Histogram of Training Time Series")
acf(cpi_train, lag.max = 80, main = "ACF of Training Time Series" )

# Box-Cox transformation
t <- 1:length(cpi_train)
fit <- lm(cpi_train ~ t)
bcTransform = boxcox(cpi_train ~ t, plotit = TRUE)
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda
cpi.bc = (1/lambda)*(cpi_train^lambda-1)

# compare variance
var(cpi_train)
var(cpi.bc)

# Plot the original data vs Box-Cox transformed data vs Log transformed data
op <- par(mfrow = c(2,2))
ts.plot(cpi_train, main = "Original Training Data", ylab = expression(X[t]))
ts.plot(cpi.bc, main = "Box-Cox Tranformed Training Data",
        ylab = expression(Y[t]))
hist(cpi_train, main = "Histogram of Original(training)", col = "purple")
hist(cpi.bc, main = "Histogram of Box-Cox(training)", col = "darkgreen")
par(op)

# Differencing (remove trend)
var(cpi.bc)
# Since there is a trend, I use difference at lag 1
cpi_train1 <- diff(cpi.bc, lag = 1)  # first difference
var(cpi_train1)

cpi_train2 <- diff(cpi_train1, lag = 1)  # second difference
var(cpi_train2)  # sample variance increases, strop differencing at lag 1

ts.plot(cpi_train1, main = "De-trended & Transformed Training Time Series",
        ylab = expression(nabla~Y[t]))
abline(h = mean(cpi_train1), lty = 2)

# Histogram with normal curve of de-trended training
hist(cpi_train1, density = 20, breaks = 20, col = "lightblue", prob = TRUE,
     main = "Histogram of De-trended & Transformed Training Time Series")
m <- mean(cpi_train1)
std <- sqrt(var(cpi_train1))
curve(dnorm(x, m, std), add = TRUE)

# Model Identification
# ACF and PACF
```

```r
op <- par(mfrow=c(1,2))
acf(cpi_train1, lag.max = 20, main = expression(nabla~Y[t]))
pacf(cpi_train1, lag.max = 20, main = expression(nabla~Y[t]))
par(op)

# Model Estimation
aiccs = matrix(NA, nr = 42, nc = 3)
colnames(aiccs) = c("p", "q", "AICc")
i = 0
for(p in 0:5){
  for(q in 0:6){
    aiccs[i+1, 1] = p
    aiccs[i+1, 2] = q
    aiccs[i+1, 3] = AICc(arima(cpi_train1, order = c(p,0,q), method = "ML"))
    i = i+1
    }
}
# 6 models with the lowest AICcs
aiccs[order(aiccs[,3])[1:12],]

length(cpi_train)  # n = 156
sqrt(length(cpi_train))

# Model Diagnostics
# ARIMA(5,1,4)
# ar1 = 0
(fit1 <- arima(cpi_train1, order = c(5,1,4), method = "ML"))
(fit1_n <- arima(cpi_train1, order = c(5,1,4), fixed = c(0,0,NA,NA,NA,NA,NA,NA,NA),
                 method = "ML"))
AICc(fit1_n)

Box.test(residuals(fit1_n), lag = 12, type ="Box-Pierce", fitdf = 10)
Box.test(residuals(fit1_n), lag = 12, type = "Ljung-Box", fitdf = 10)
Box.test((residuals(fit1_n))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
# McLeod-Li test: Ljung-Box for squares
shapiro.test(residuals(fit1_n))

# ARIMA(5,1,5)
(fit2 <- arima(cpi_train1, order = c(5,1,5), method = "ML"))
(fit2_n <- arima(cpi_train1, order = c(5,1,5),
                 fixed = c(NA,NA,0,NA,NA,NA,NA,NA,NA,NA), method = "ML"))
AICc(fit2_n)

Box.test(residuals(fit2_n), lag = 12, type ="Box-Pierce", fitdf = 11)
# p-value < 0.05, so it fails to pass Box-Pierce test.
Box.test(residuals(fit2_n), lag = 12, type = "Ljung-Box", fitdf = 11)
# p-value < 0.05, so it fails to pass Ljung-Box test.
Box.test((residuals(fit2_n))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
# McLeod-Li test: Ljung-Box for squares
shapiro.test(residuals(fit2_n))

# ARIMA(2,1,4)
(fit3 <- arima(cpi_train1, order = c(2,1,4), method = "ML"))
```

```r
(fit3_n <- arima(cpi_train1, order = c(2,1,4), fixed = c(NA,NA,0,0,NA,0),
                method = "ML"))
AICc(fit3_n)

Box.test(residuals(fit3_n), lag = 12, type ="Box-Pierce", fitdf = 7)
Box.test(residuals(fit3_n), lag = 12, type = "Ljung-Box", fitdf = 7)
Box.test((residuals(fit3_n))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
shapiro.test(residuals(fit3_n))

# ARIMA(2,1,3)
# ar1 = 0
(fit4 <- arima(cpi_train1, order = c(2,1,3), method = "ML"))
(fit4_n <- arima(cpi_train1, order = c(2,1,3), fixed = c(0,NA,NA,NA,NA),
                method = "ML"))
AICc(fit4_n)

Box.test(residuals(fit4_n), lag = 12, type ="Box-Pierce", fitdf = 6)
Box.test(residuals(fit4_n), lag = 12, type = "Ljung-Box", fitdf = 6)
Box.test((residuals(fit4_n))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
shapiro.test(residuals(fit4_n))

# ARIMA(5,1,6)
(fit5 <- arima(cpi_train1, order = c(5,1,6), method = "ML"))
(fit5_n <- arima(cpi_train1, order = c(5,1,6),
                fixed = c(0,NA,NA,NA,NA,NA,NA,NA,NA,NA,NA), method = "ML"))
AICc(fit5_n)

Box.test(residuals(fit5_n), lag = 12, type ="Box-Pierce", fitdf = 12)
# p-value < 0.05, so it fails to pass Box-Pierce test.
Box.test(residuals(fit5_n), lag = 12, type = "Ljung-Box", fitdf = 12)
# p-value < 0.05, so it fails to pass Ljung-Box test.
Box.test((residuals(fit5_n))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
shapiro.test(residuals(fit5_n))

# ARIMA(3,1,4)
(fit6 <- arima(cpi_train1, order = c(3,1,4), method = "ML"))
(fit6_n <- arima(cpi_train1, order = c(3,1,4), fixed = c(NA,NA,0,NA,0,NA,NA),
                method = "ML"))
AICc(fit6)

Box.test(residuals(fit6), lag = 12, type ="Box-Pierce", fitdf = 8)
Box.test(residuals(fit6), lag = 12, type = "Ljung-Box", fitdf = 8)
Box.test((residuals(fit6))^2, lag = 12, type = "Ljung-Box", fitdf = 0)
shapiro.test(residuals(fit6))

# Check Stationarity and Invertibility
# Roots checking
uc.check(pol_ = c(1,0,0,-0.6130,0), plot_output = TRUE)  # MA part
uc.check(pol_ = c(1,-0.4954,-0.0916,0.4130), plot_output = TRUE)  # AR part

# Residual Analysis
res3 = residuals(fit3_n)
ts.plot(res3, main = "Fitted Residuals for fixed ARIMA (2,1,4)")
```

```r
t = 1:length(res3)
fit.res3 = lm(res3~t)
abline(fit.res3)
abline(h = mean(res3), col = "red")

par(mfrow=c(1,2), oma=c(0,0,2,0))
op <- par(mfrow=c(2,2))
acf(res3, main = "Autocorrelation")
pacf(res3, main = "Partial Autocorrelation")
hist(res3, main = "Histogram")
qqnorm(res3)
qqline(res3, col ="blue")
title("Fitted Residuals Diagnostics for fixed ARIMA (2,1,4)", outer = TRUE)
par(op)

# Roots checking
uc.check(pol_ = c(1,-0.5257,0,-0.6262,-0.0668), plot_output = TRUE)  # MA part
uc.check(pol_ = c(1,-0.8756,0.2637,-0.3881), plot_output = TRUE)  # AR part

# Spectral Analysis
require(TSA)
periodogram(res3)
abline(h = 0)
# Fisher's test
fisher.g.test(res3)
# Kolmogorov Smirnov Test
cpgram(res3, main = "")

# Forecasting
# Predict 12 future observations with transformed time series Y and the plot
# candidate model: new ARIMA(2,1,4)
fit3_n = arima(cpi.bc, order = c(2,1,4), fixed = c(NA,NA,0,0,NA,0), method = "ML")
pred.tr <- predict(fit3_n, n.ahead = 12)
U.tr = pred.tr$pred + 1.96*pred.tr$se
# upper bound for the prediction interval for transformed data
L.tr = pred.tr$pred - 1.96*pred.tr$se  # lower bound
ts.plot(cpi.bc, xlim=c(1,length(cpi_train)+12), ylim = c(min(cpi.bc),max(U.tr)),
        main="Prediction on Transformed Training Time Series")
lines(U.tr, col = "blue", lty = "dashed")
lines(L.tr, col = "blue", lty = "dashed")
points((length(cpi_train)+1):(length(cpi_train)+12), pred.tr$pred, col = "red")

# Get predictions and s.e.'s of original time series X
pred.orig <- (-2*pred.tr$pred+1)^(-1/2)
# back-transform to get predictions of original time series
# bounds of the prediction interval
U = (-2*U.tr+1)^(-1/2)
L = (-2*L.tr+1)^(-1/2)

# Predict 12 future observations with original training data
ts.plot(cpi_train, xlim=c(1,length(y)+12), ylim = c(min(cpi_train),max(U)),
        main="Prediction on Original Training Time Series")
lines(U, col="blue", lty="dashed")
```

```
lines(L, col="blue", lty="dashed")
points((length(cpi_train)+1):(length(cpi_train)+12), pred.orig, col="red")
# points((length(cpi_train)+1):(length(cpi_train)+12), cpi_test, col = "green")


# Plot the last 12 values plus forecast
ts.plot(cpi_train, xlim=c(length(cpi_train)-12,length(cpi_train)+12),
        ylim = c(min(cpi_train),max(U)),
        main="Zoom Graph of Prediction on Original Training Time Series")
points((length(cpi_train)+1):(length(cpi_train)+12), pred.orig, col="red")
points((length(cpi_train)+1):(length(cpi_train)+12), cpi_test, col = "green")
lines((length(cpi_train)+1):(length(cpi_train)+12), U, lty=2, col="blue")
lines((length(cpi_train)+1):(length(cpi_train)+12), L, lty=2, col="blue")
```