ECON 430 - Project 2

Benedikt Graf (105652212); Wong Jing Hei (805643634); Qiumeng He (305524290)

11-18-2020

Contents

I. Introduction

With the passage of the Federal Reserve Reform Act in 1977, the Federal Reserve was directed by Congress to "promote effectively the goals of maximum employment, stable prices, and moderate long term interest rates" (Federal Reserve Reform Act of 1977). Inflation and unemployment are critical metrics in evaluating economic development, national economic stability, and living standards. The Fed's objective of minimizing unemployment and stabilizing price levels is commonly known as the "dual mandate" of the Federal Reserve (Federal Reserve Board of Governors, 2018). There exists a natural tension in this mandate which can be observed through the Phillips Curve, which relates the inflation rate to the unemployment rate. Low unemployment drives inflation up and vice versa (see Ng, Wessel, and Sheiner, 2018). Over the course of the 20th century, it was questioned whether the Federal Reserve has been upholding both parts of its dual mandate. Such was the case in the 1980s when "the Federal Reserve pursued an aggressive set of policies designed to reduce inflation" and unemployment rose (Steelman 2011, p.3). However, recently the relationship between inflation and unemployment seems to have weakened. Binder (2018) explains that "since 2000, the correlation between unemployment and changes in inflation is nearly zero."

For this project, we decided to investigate this relationship by analyzing the time series of two macroeconomic indicators that track the dual mandate of the Fed: Consumer Price Index (CPI) and the Unemployment Rate (UR). The Consumer Price Index measures the price level changes of consumer goods and services purchased by households and is a common measure of inflation.[1] We retrieved these data from FRED at the Federal Reserve Bank of St. Louis.

We first analyze the two time-series individually by analyzing their ACF and PACF plots, testing their stationarity, and fitting different models to describe their trend, seasonality, and cycles. Then we investigate the causal relationship between the two macroeconomic variables using a Vector Autoregression (VAR) model to make effective forecasts. Finally, we will attempt to judge the impact of COVID-19 on the causal relationship between the two series.

[1]We choose the Sticky Price Consumer Price Index less Food and Energy rather than the Consumer Price Index, because the fluctuation of the former index more effectively reflect the changes of people's living standards.

Preliminary Data Analysis

```r
# Overivew of the data
cat("Number of rows:   ", dim(data)[1], "\n")
```

```
## Number of rows:    729
```

```r
cat("Number of columns:", dim(data)[2])
```

```
## Number of columns: 3
```

```r
# Checking for Missing Values
sum(is.na(data))
```

```
## [1] 0
```

```r
colSums(is.na(data))
```

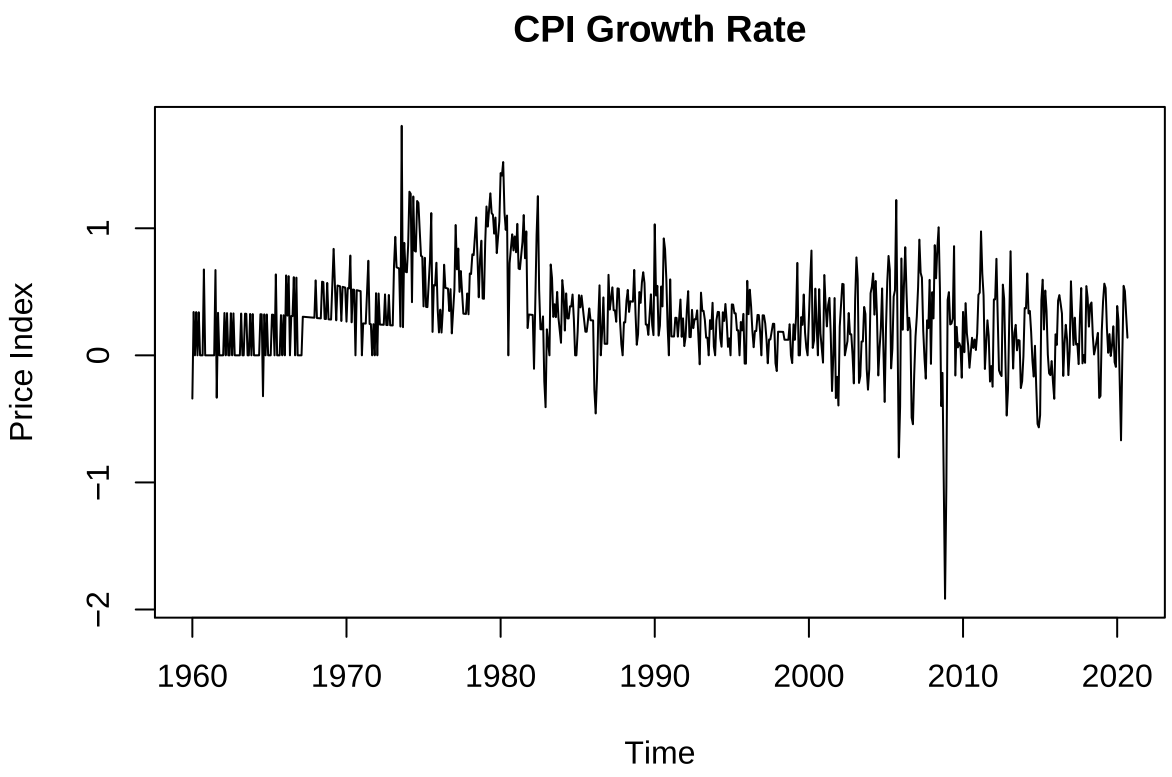
```
##   DATE UNRATE  CPIGR
##      0      0      0
```

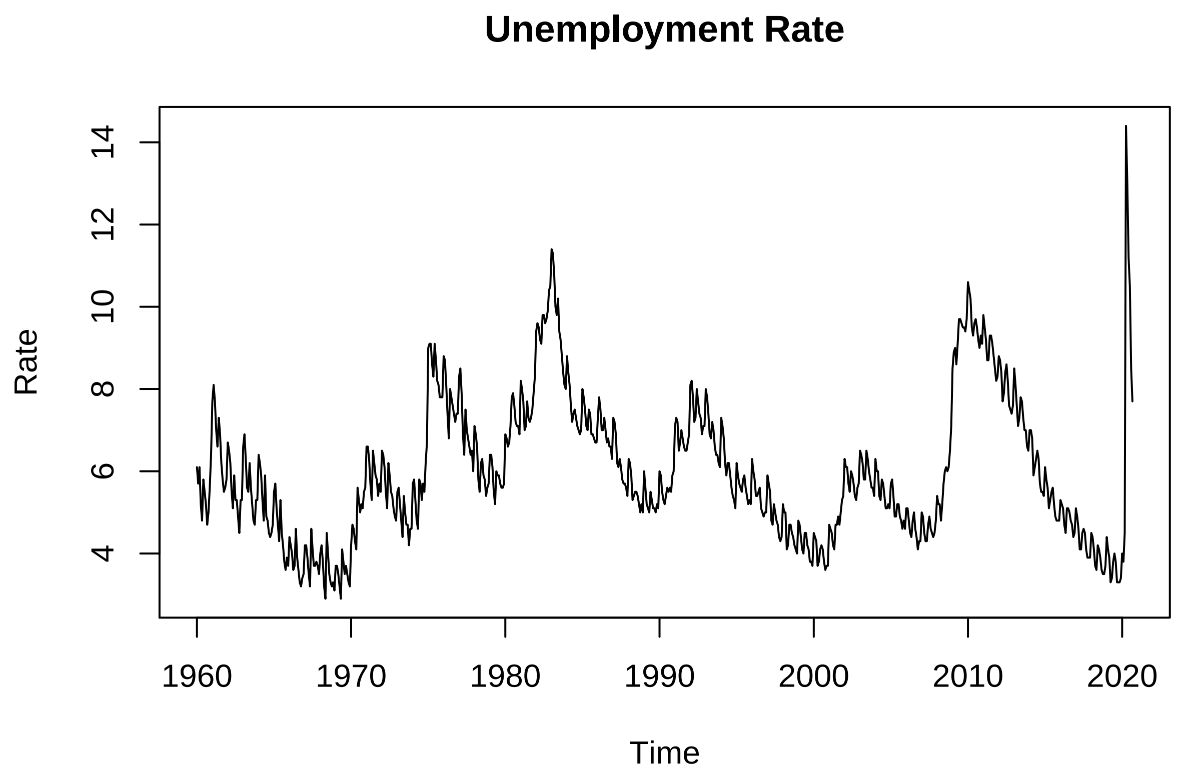
No missing data, therefore, no imputations are needed.

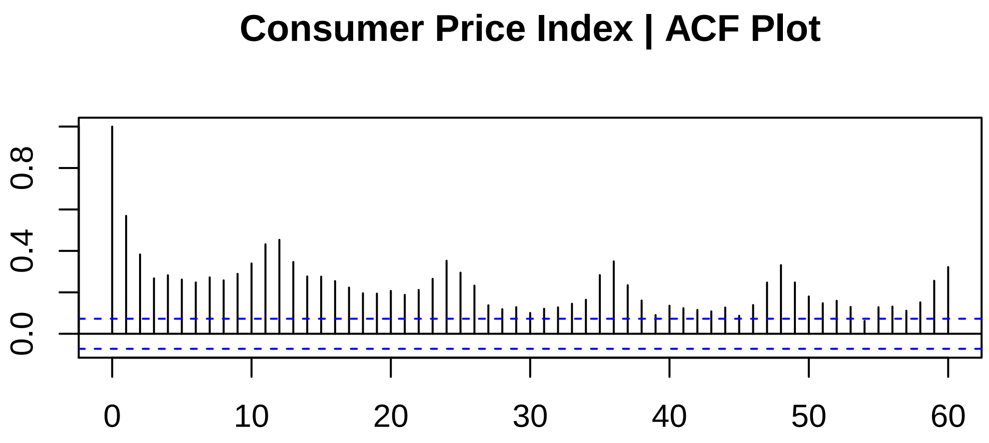
II. Results - Modeling and Forecasting Trend, Seasonality, and Cycles

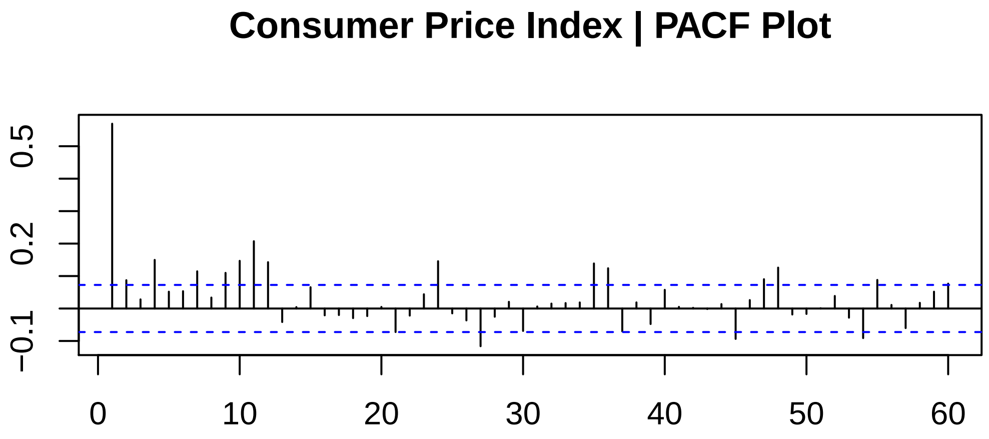
Time Series Plot with ACF/PACF

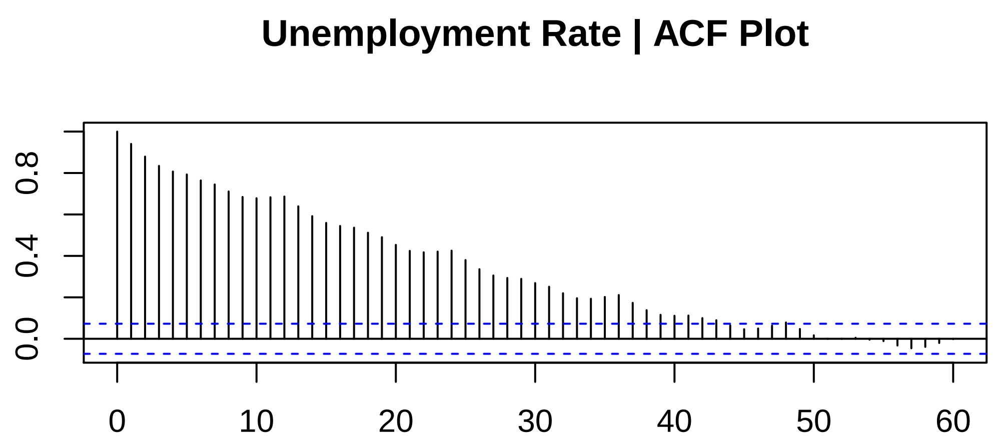
Produce a time-series plot of your data including the respective ACF and PACF plots.

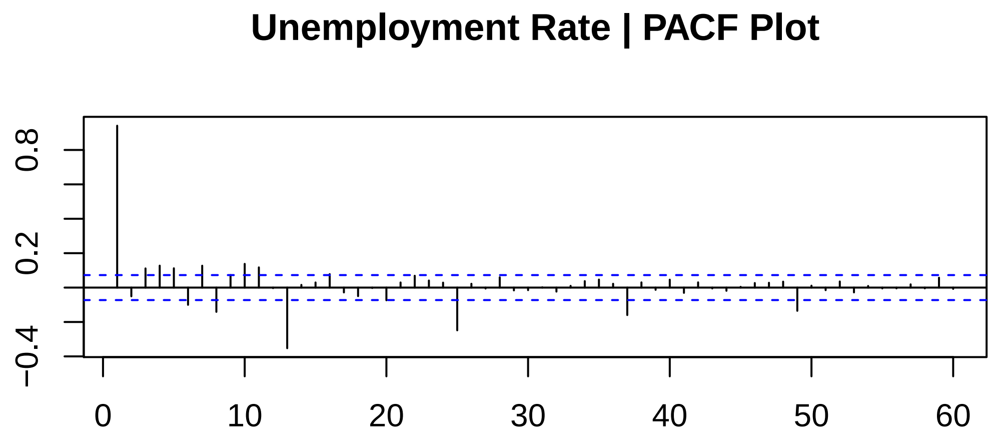

CPI Growth Rate
Price Index
Time
1960
1970
1980
1990
2000
2010
2020
−2
−1
0
1


Unemployment Rate
Rate
14
12
10
8
6
4
1960
1970
1980
1990
2000
2010
2020
Time


Consumer Price Index | ACF Plot
0.0
0.4
0.8
0
10
20
30
40
50
60


Consumer Price Index | PACF Plot
−0.1
0.2
0.5
0
10
20
30
40
50
60





Auto ARIMA Model Fit

As a baseline model, fit an ARIMA model to each series and comment on the fit. For the next questions, you will instead use the model estimated in (3) for their respective answers.

```
# Auto ARIMA models
ARIMA_cpi_auto <- auto.arima(CPIGRL_ts)
ARIMA_cpi_auto
```

```
## Series: CPIGRL_ts
## ARIMA(3,1,5)(0,0,1)[12] with drift
##
## Coefficients:
##           ar1      ar2      ar3      ma1     ma2      ma3      ma4      ma5
##       -0.5003  -0.9127  -0.1713  -0.0661  0.4397  -0.6448  -0.3012  -0.2253
## s.e.   0.2559   0.2681   0.2288   0.2512  0.2090   0.1213   0.2083   0.0957
##          sma1   drift
##        0.1852  0.0001
## s.e.   0.0367  0.0010
##
## sigma^2 estimated as 0.07663:  log likelihood=-93.88
## AIC=209.75   AICc=210.12   BIC=260.24
```

```r
ARIMA_ur_auto <- auto.arima(UNRATE_ts)
ARIMA_ur_auto
```

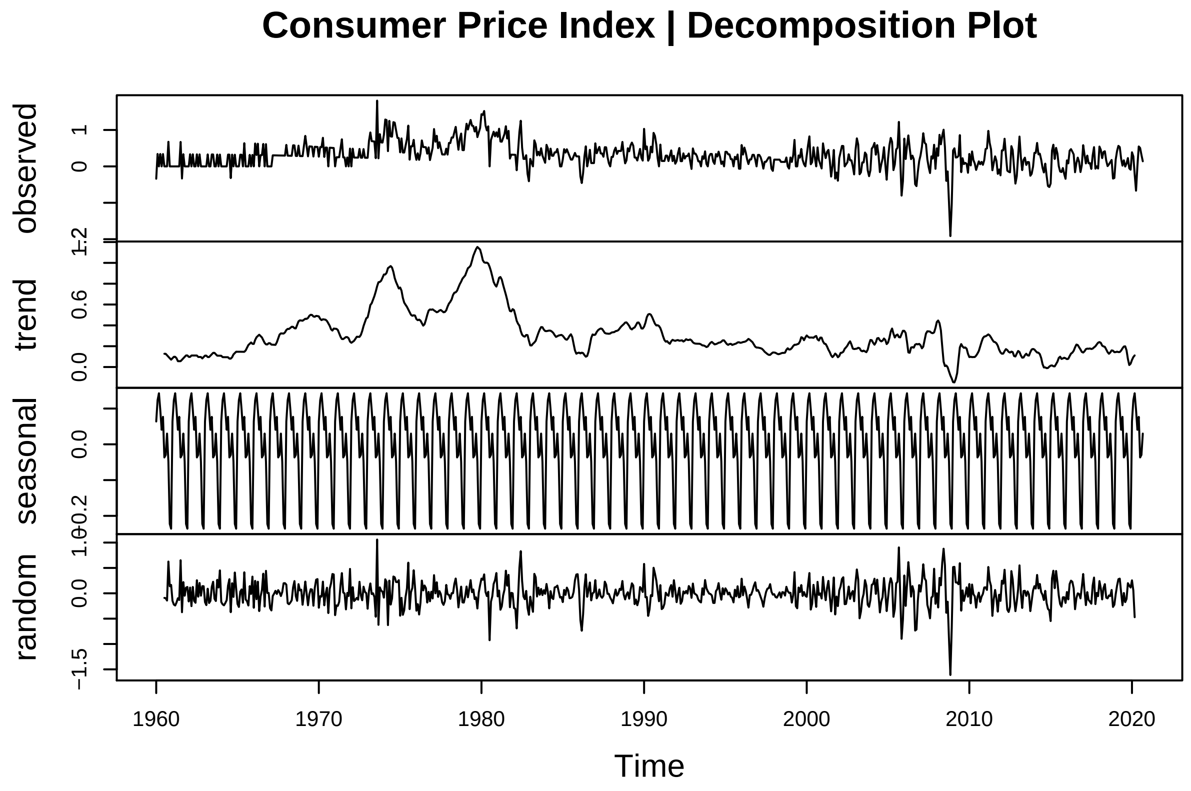
```
## Series: UNRATE_ts
## ARIMA(2,1,0)(1,0,0)[12]
##
## Coefficients:
##          ar1      ar2    sar1
##       0.0205  -0.0645  0.7507
## s.e.  0.0370   0.0377  0.0383
##
## sigma^2 estimated as 0.2276:  log likelihood=-497.61
## AIC=1003.22   AICc=1003.28   BIC=1021.58
```

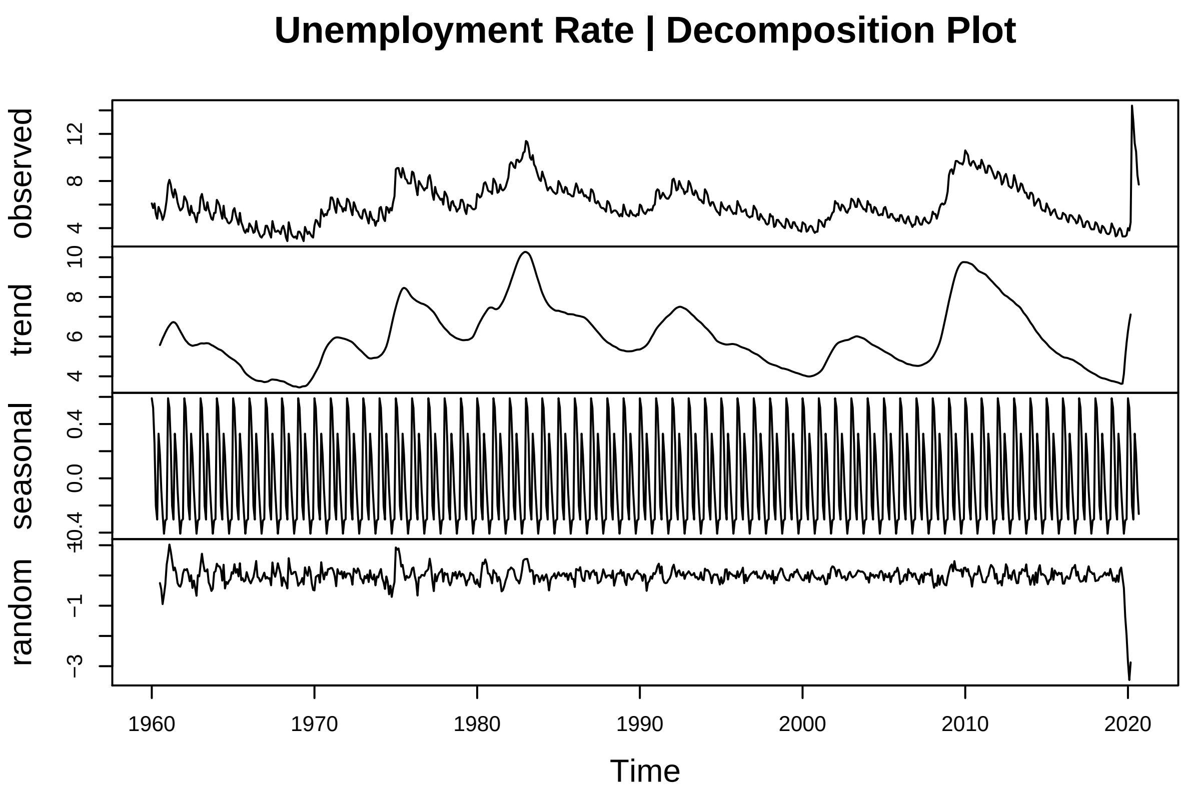
The auto.arima function proposes an ARIMA(3,1,5)(0,0,1)[12] for the CPI and an ARIMA (2,1,0)(1,0,0)[12] for the UR. It seems that some of the suggested components are not significant (e.g. the third AR lag for CPI), so we hope to improve on these benchmark models in part 1(3).

Seasonal Decomposition + Manual ARIMA Fit

Fit a model that includes, trend, seasonality and cyclical components. Make sure to discuss your model in detail.

To get a better understanding of the series' components we take a look at the decomposition plots first.


Consumer Price Index | Decomposition Plot
observed
trend
seasonal
random
Time
1960
1970
1980
1990
2000
2010
2020



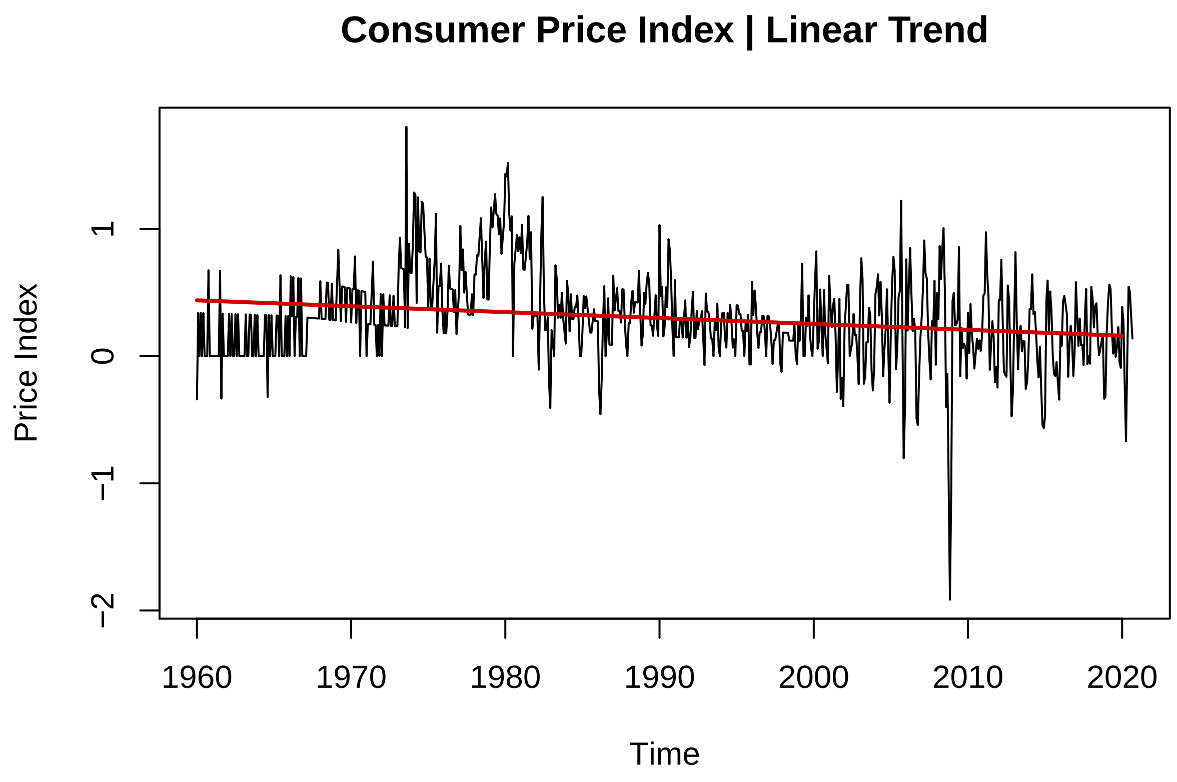
These plots show us that there is evidence of trends, seasons, and cycles in both of our time series. The trend component seems highly irrgeular especially for the CPI. We will now analyze each of these components in turn.

Trend

We fit different trends to our series below.

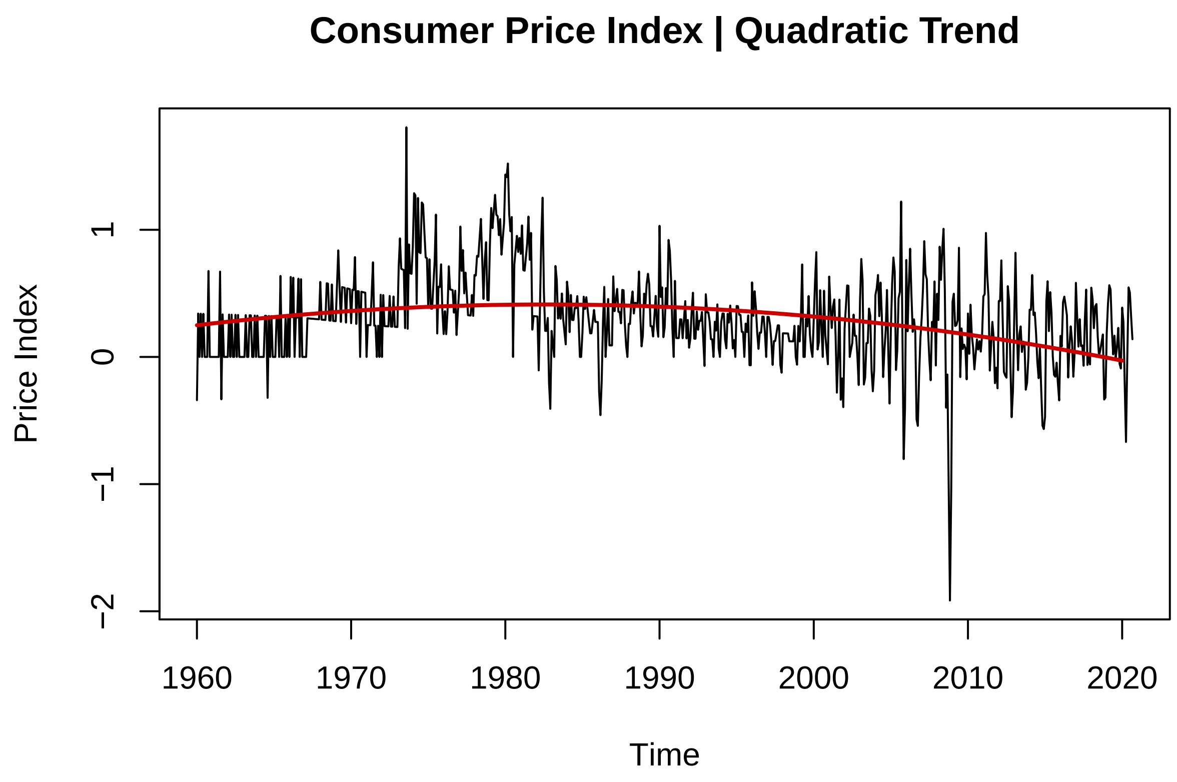
```
# Creating a variable for the time trend
t <- seq(1960,2020,length=length(CPIGRL_ts))
##t = seq_len(length(CPIGRL_ts))
t2 = t^2
t3 = t^3
t4 = t^4
```

CPI Linear Trend



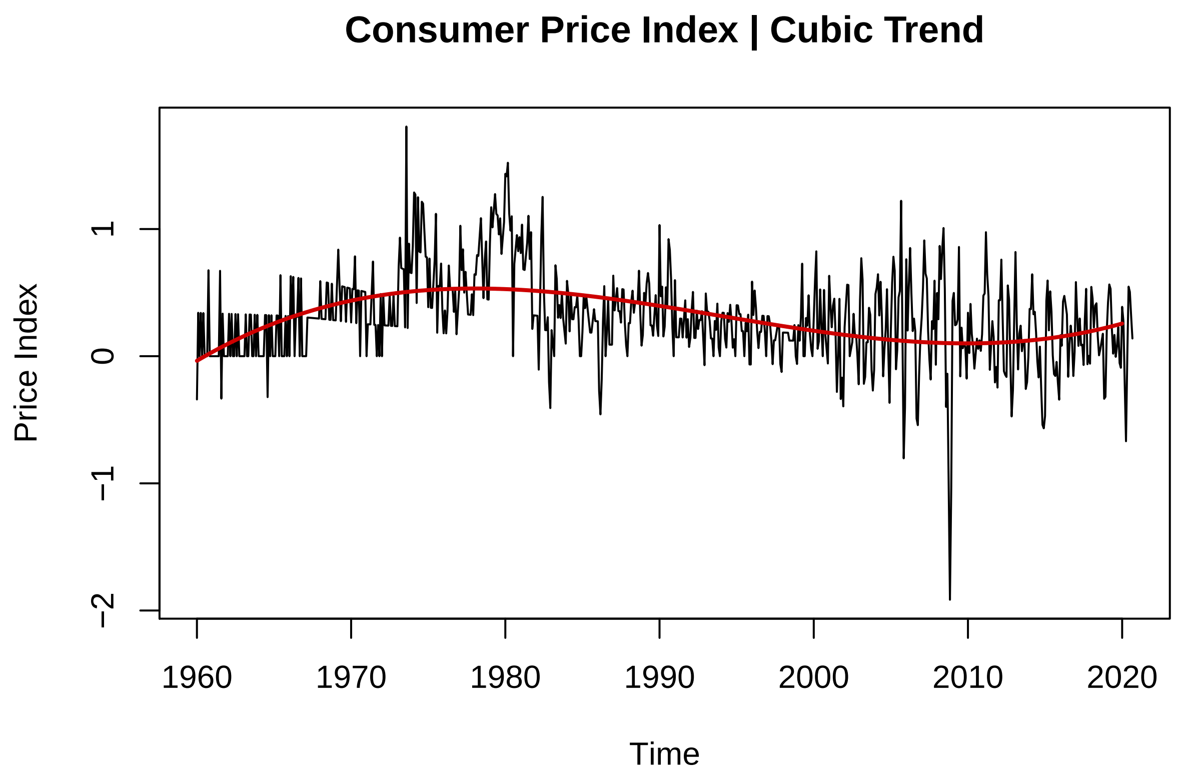
```
##
## Call:
## lm(formula = CPIGRL_ts ~ t)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.13061 -0.18129 -0.02955  0.18725  1.42877
##
## Coefficients:
##               Estimate Std. Error t value       Pr(>|t|)
## (Intercept)  9.5345174  1.4736064   6.470 0.000000000180 ***
## t           -0.0046403  0.0007405  -6.267 0.000000000632 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3468 on 727 degrees of freedom
## Multiple R-squared:  0.05125,    Adjusted R-squared:  0.04995
## F-statistic: 39.27 on 1 and 727 DF,  p-value: 0.0000000006316
```

CPI Quadratic Trend



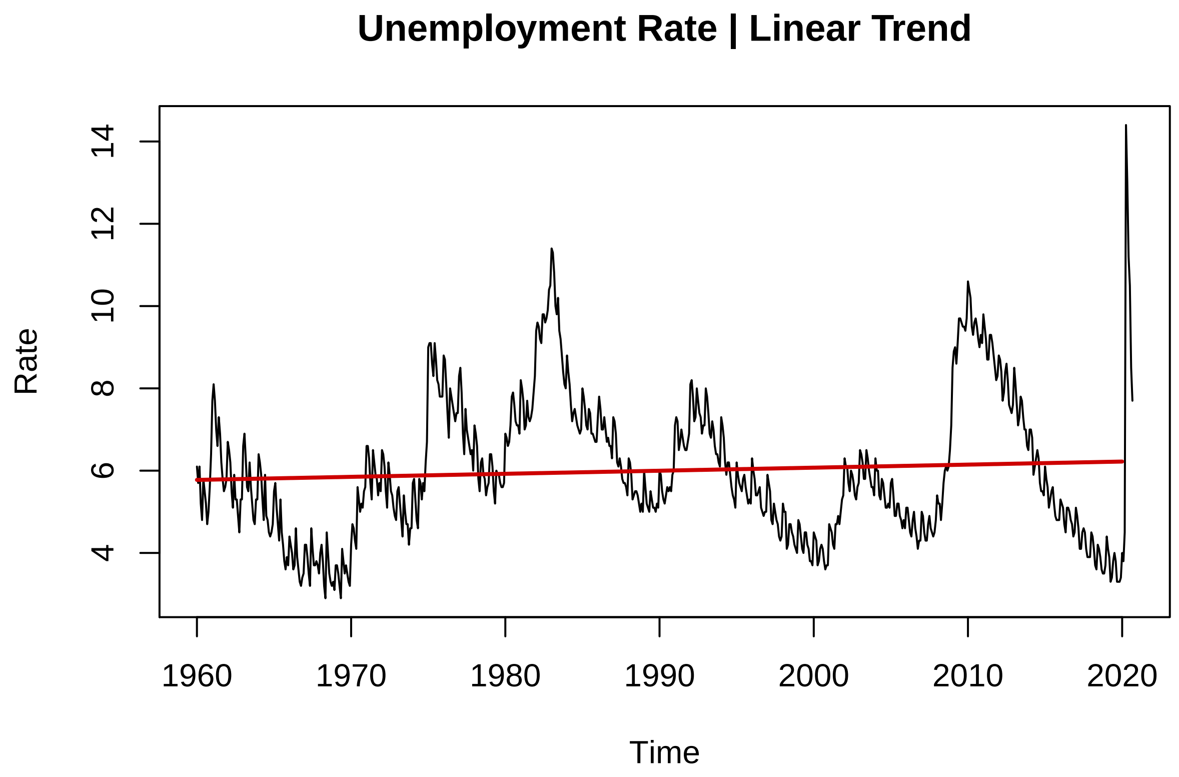
```
##
## Call:
## lm(formula = CPIGRL_ts ~ t + t2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.11985 -0.22486 -0.03595  0.19600  1.42041
##
## Coefficients:
##                    Estimate     Std. Error t value        Pr(>|t|)
## (Intercept) -1245.81151118   183.31777562  -6.796 0.0000000000225 ***
## t               1.25710980     0.18424877   6.823 0.0000000000188 ***
## t2             -0.00031702     0.00004629  -6.848 0.0000000000160 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3363 on 726 degrees of freedom
## Multiple R-squared:  0.1088, Adjusted R-squared:  0.1064
## F-statistic: 44.32 on 2 and 726 DF,  p-value: < 0.00000000000000022
```

CPI Cubic Trend



```
##
## Call:
## lm(formula = CPIGRL_ts ~ t + t2 + t3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.01941 -0.20058 -0.03119  0.18267  1.30292
##
## Coefficients:
##                       Estimate        Std. Error t value            Pr(>|t|)
## (Intercept) -210495.087214058   22667.079614879  -9.286 <0.0000000000000002 ***
## t               316.737038498      34.174035787   9.268 <0.0000000000000002 ***
## t2               -0.158856876       0.017173497  -9.250 <0.0000000000000002 ***
## t3                0.000026556       0.000002877   9.232 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3183 on 725 degrees of freedom
## Multiple R-squared:  0.2026, Adjusted R-squared:  0.1993
## F-statistic: 61.39 on 3 and 725 DF,  p-value: < 0.00000000000000022
```

UR Linear Trend

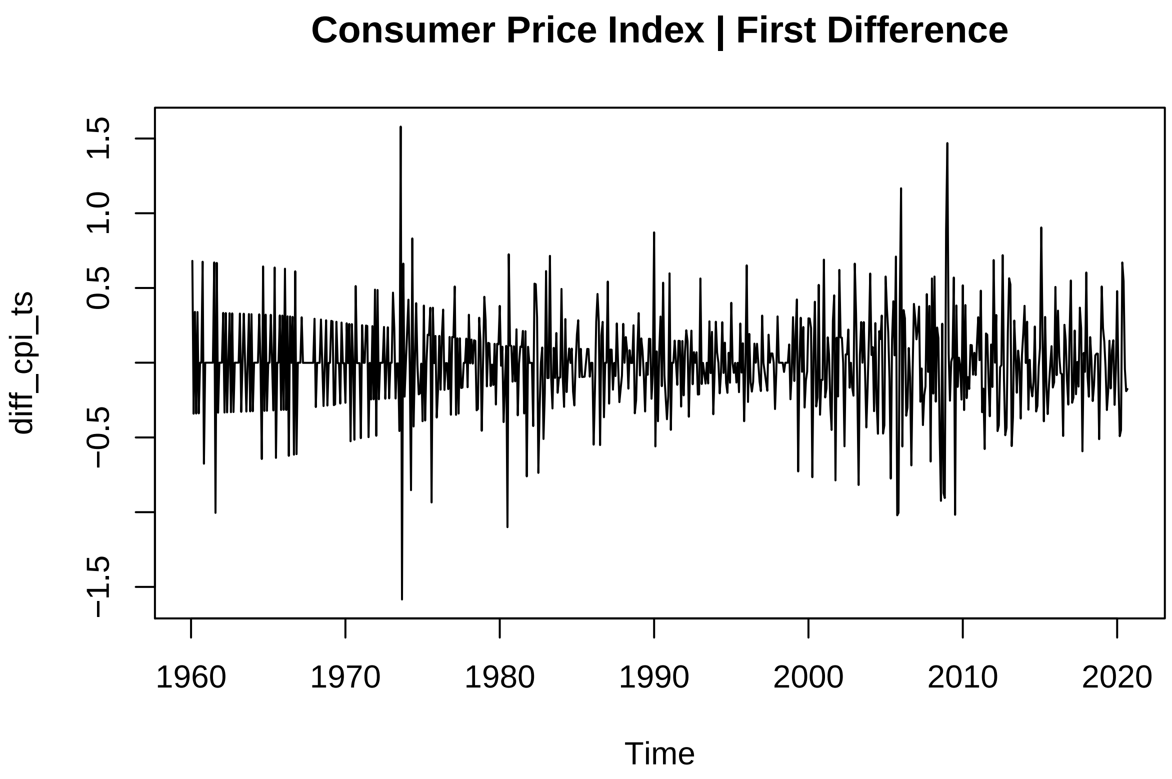


```
##
## Call:
## lm(formula = UNRATE_ts ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9442 -1.2518 -0.3001  1.0413  8.1817
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.784131   7.294622  -1.204   0.2289
## t            0.007428   0.003666   2.027   0.0431 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.717 on 727 degrees of freedom
## Multiple R-squared:  0.005618,   Adjusted R-squared:  0.00425
## F-statistic: 4.107 on 1 and 727 DF,  p-value: 0.04307
```

Although, the cubic trend fits the CPI best, the linear trend makes more economic sense. The Unemployment Rate seems to be relatively trendless.

Seasonality

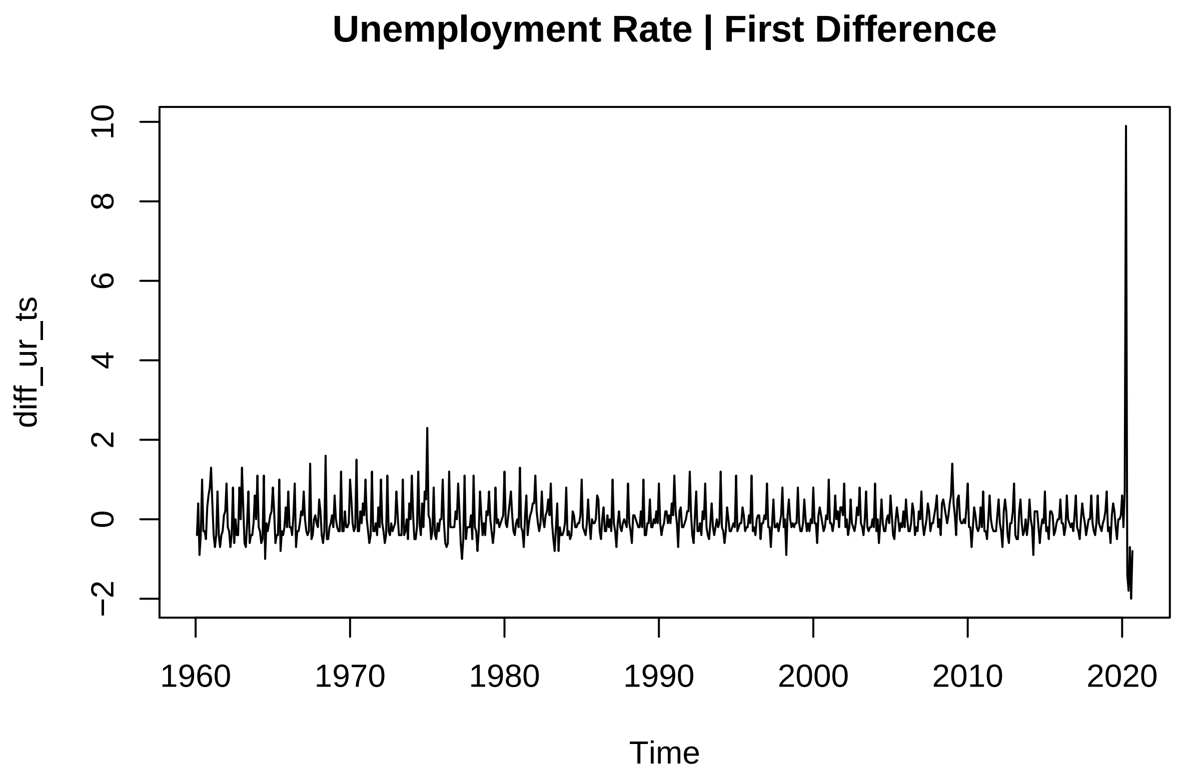
Before we fit seasonal models to the data, we check whether our series are stationary or non-stationary. This will help us in adding seasonal components and later when we fit ARIMA models.



```
## Warning in adf.test(diff_cpi_ts): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_cpi_ts
## Dickey-Fuller = -16.357, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

For the CPI we reject the null and accept the alternative hypothesis and can therefore say that the series is stationary.

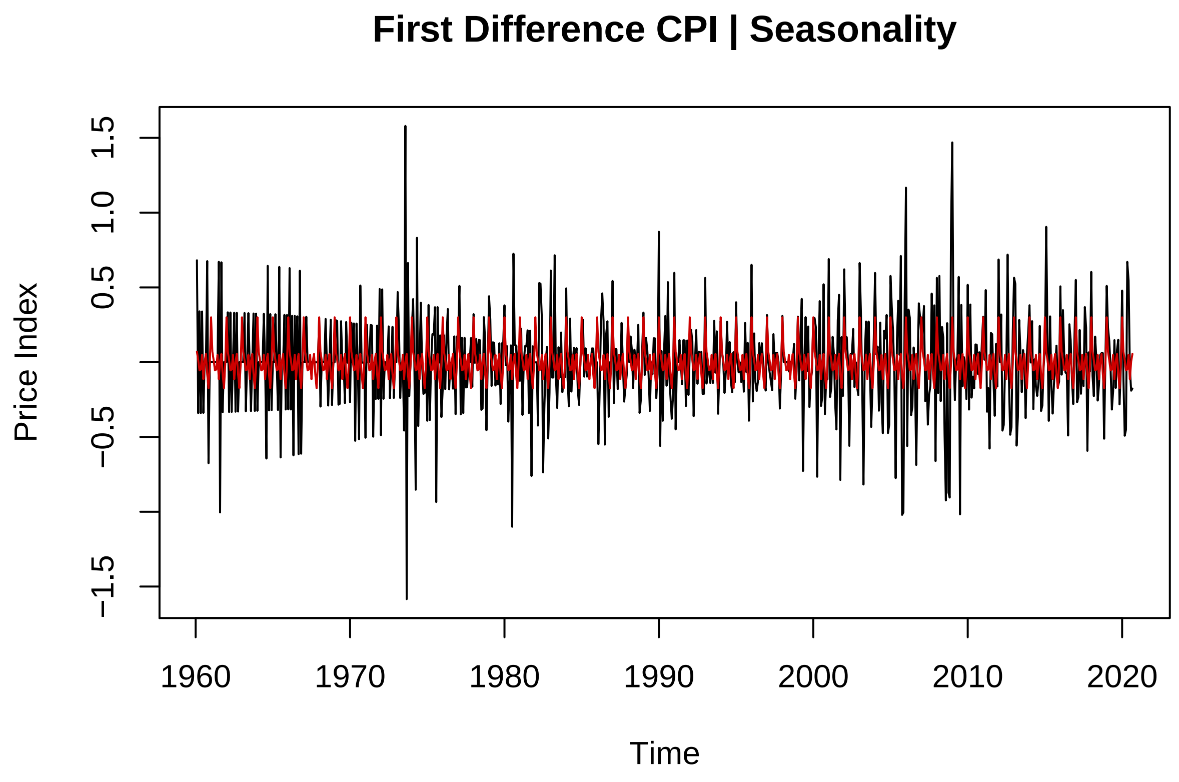


```
##
##  Augmented Dickey-Fuller Test
##
## data:  UNRATE_ts
## Dickey-Fuller = -3.1878, Lag order = 8, p-value = 0.09003
## alternative hypothesis: stationary
```

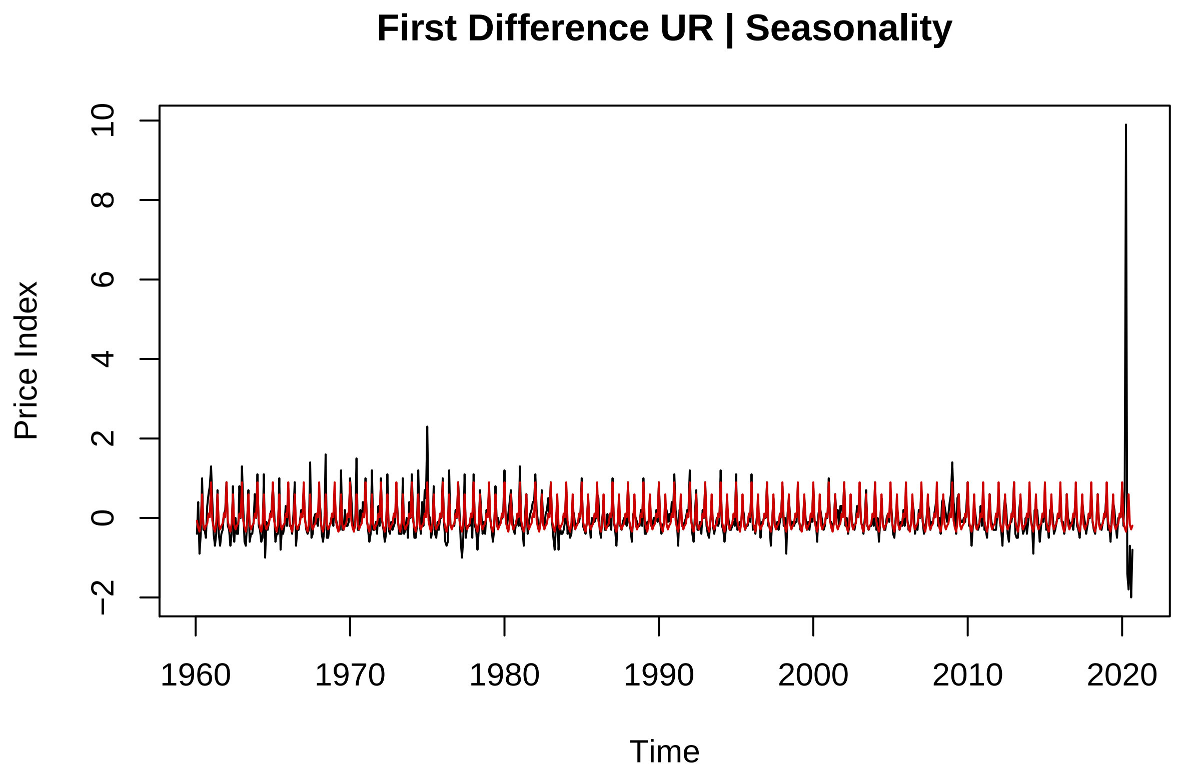
```
## Warning in adf.test(diff_ur_ts): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_ur_ts
## Dickey-Fuller = -11.107, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

For the unemployment rate, the ADF test of the series suggest that the series is non-stationary (fail to reject null). We can cofirm this by applying the ADF test to the first-difference of the UR: we reject the null-hypothesis that the first-difference of the series is non-stationary, and accept the alternative hypothesis that it is stationary. The first-difference of the UR series is stationary I(O), so we conclude that the series itslef is non-stationary I(1).



```
##
## Call:
## tslm(formula = diff_cpi_ts ~ 0 + season)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.63970 -0.17639  0.00844  0.19078  1.57526
##
## Coefficients:
##           Estimate Std. Error t value          Pr(>|t|)
## season1   0.300202   0.040233   7.462 0.000000000000248 ***
## season2   0.070434   0.039901   1.765           0.07796 .
## season3   0.013706   0.039901   0.343           0.73133
## season4  -0.054675   0.039901  -1.370           0.17104
## season5  -0.048842   0.039901  -1.224           0.22133
## season6   0.049356   0.039901   1.237           0.21651
## season7  -0.114359   0.039901  -2.866           0.00428 **
## season8   0.004366   0.039901   0.109           0.91291
## season9   0.055562   0.039901   1.392           0.16421
## season10 -0.079820   0.040233  -1.984           0.04764 *
## season11 -0.174361   0.040233  -4.334 0.00001675037148 ***
## season12 -0.013171   0.040233  -0.327           0.74348
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3116 on 716 degrees of freedom
## Multiple R-squared:  0.1191, Adjusted R-squared:  0.1043
## F-statistic: 8.067 on 12 and 716 DF,  p-value: 0.00000000000002615
```

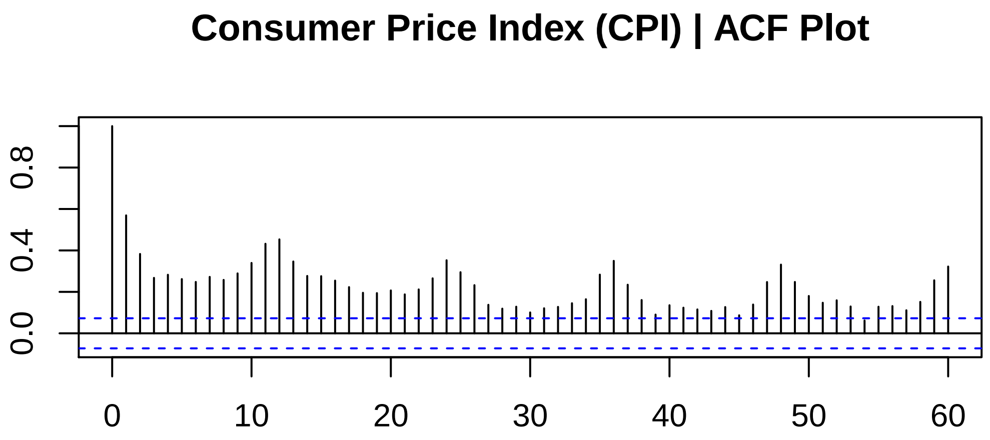


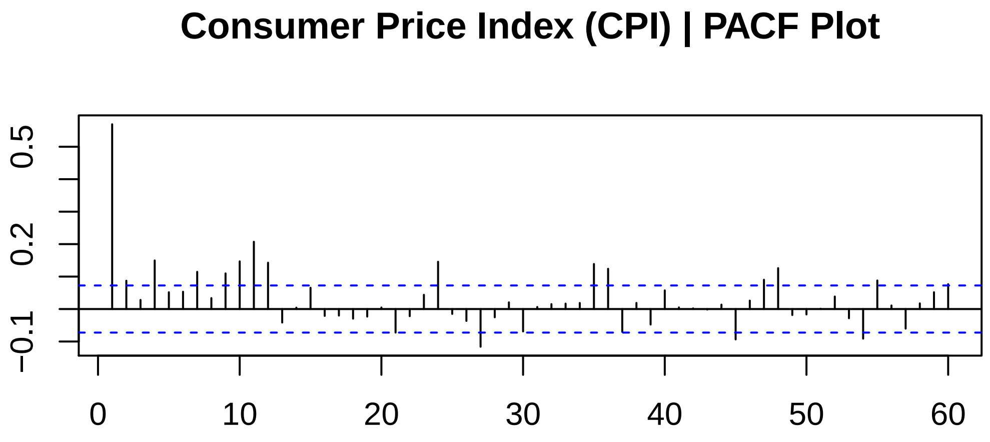
```
##
## Call:
## tslm(formula = diff_ur_ts ~ 0 + season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3951 -0.1541 -0.0183  0.1049 10.2459
##
## Coefficients:
##          Estimate Std. Error t value             Pr(>|t|)
## season1   0.89833    0.06077  14.783 < 0.0000000000000002 ***
## season2  -0.07049    0.06027  -1.170              0.24253
## season3  -0.24754    0.06027  -4.107          0.000044643 ***
## season4  -0.34590    0.06027  -5.739          0.000000014 ***
## season5  -0.12951    0.06027  -2.149              0.03198 *
## season6   0.59508    0.06027   9.874 < 0.0000000000000002 ***
## season7  -0.16557    0.06027  -2.747            0.00616 **
## season8  -0.28852    0.06027  -4.787        0.000002054 ***
## season9  -0.19344    0.06027  -3.210            0.00139 **
## season10 -0.14000    0.06077  -2.304            0.02152 *
## season11  0.11000    0.06077   1.810            0.07069 .
## season12  0.01833    0.06077   0.302            0.76297
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4707 on 716 degrees of freedom
## Multiple R-squared:  0.3704, Adjusted R-squared:  0.3599
## F-statistic: 35.11 on 12 and 716 DF,  p-value: < 0.00000000000000022
```

There are some seasonal components present in both CPI and UR. The seasonality is much stronger for the unemployment rate than CPI, which is what we woudl expect from the literature (***REFERENCE***).

Cycles

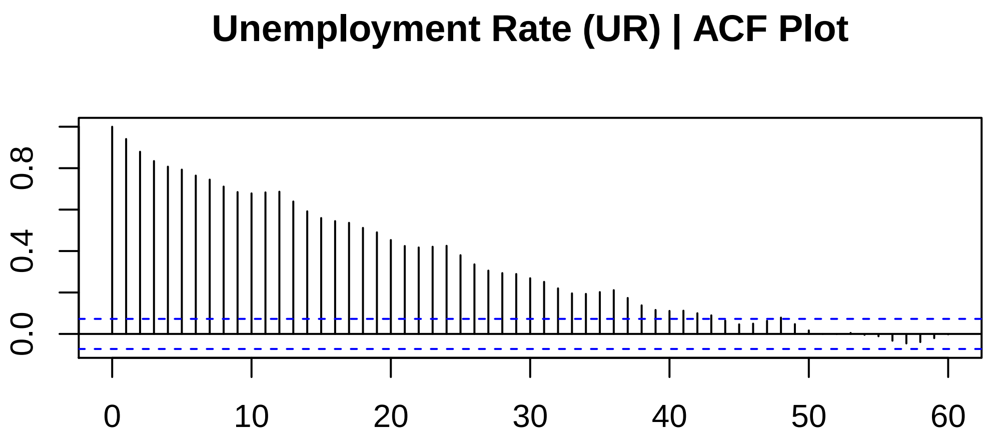
We are repeating the ACF and PACF Plots here for reference.

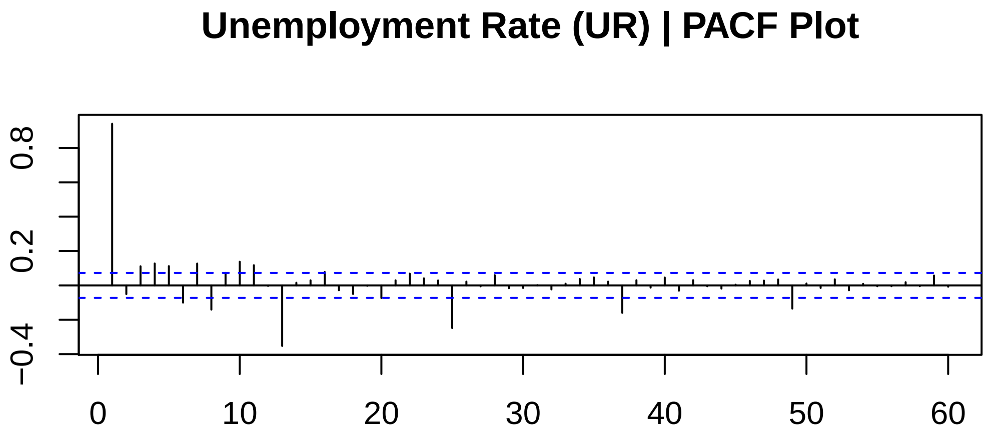




The ACF plot of CPI is gradually declining, but also exhibits a seasonal pattern. There are at least two spikes in the PACF plot at lags 1 and 2and recurring spikes at approximately 10 lags.

Therefore we suggest a seasonal-AR(2) model for the CPI series to begin. In our trend analysis above, we received some significant coefficients, so we also add a linear trend to the CPI series.





For the UR, the ACF is also decaying. There may be some seasonal pattern here too. The PACF plot exhibits one significant jump at the first lag adn then again recurring lages approximatley 10 lags apart, which suggest that seasonality is present. We begin by fitting a seasonal-AR(1) model to the UR series. The trend component for the UR was not highly significant in our analysis above, but the data were not found to be stationary, so we integrate through the first order.

ARIMA Models First Iteration

```
ARIMA_cpi_1 <- arima(CPIGRL_ts, order=c(2,0,0),seasonal=list(order=c(1,0,0)), xreg = cbind(t))
coeftest(ARIMA_cpi_1)
```

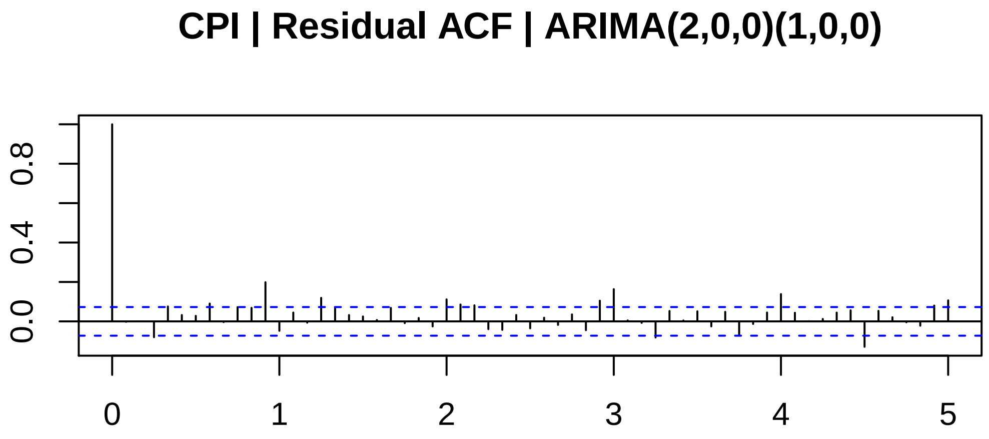
```
##
## z test of coefficients:
##
##               Estimate Std. Error z value              Pr(>|z|)
## ar1          0.4391500  0.0385771 11.3837 < 0.00000000000000022 ***
## ar2          0.0686604  0.0370840  1.8515              0.064100 .
## sar1         0.2623515  0.0384216  6.8282     0.000000000008597 ***
## intercept  8.9453354  3.2082655  2.7882              0.005300 **
## t         -0.0043458  0.0016125 -2.6951              0.007038 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

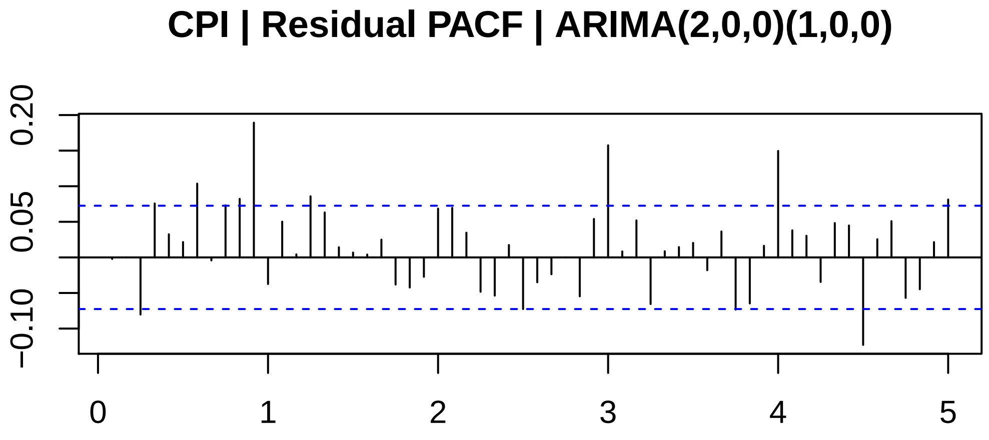
```
ARIMA_ur_1 <- arima(UNRATE_ts, order=c(1,1,0),seasonal=list(order=c(1,0,0)))
coeftest(ARIMA_ur_1)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value            Pr(>|z|)
## ar1  0.019464   0.037106  0.5246              0.5999
## sar1 0.758642   0.037583 20.1860 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

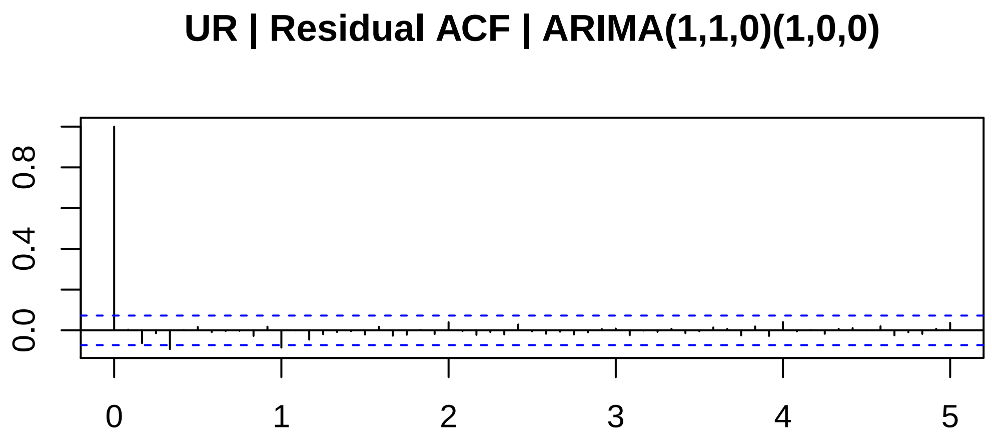
In the CPI model, all of the components are statistically significant. In the UR model, the first auto-lag is not significant.

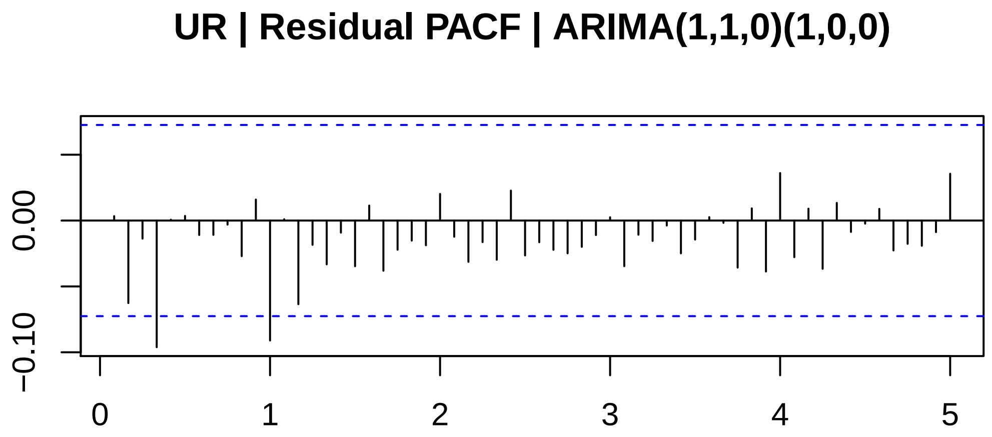
We next take a look at the ACF and PACF of the residuals to see if we can improve our model further.





In the CPI residuals, there is evidence of an MA component, as well as seasonality that is not yet included in the model. So we add an MA component and a seasonl MA component and propose ARIMA(2,0,1)SAR(1,0,1) + Trend.





The ACF and PACF plots for unemployment on the other hand do not show trends of seasonality. There is a spike in the ACF plot, so an MA component will be added: ARIMA(1,1,1)SAR(1).

ARIMA Models Second Iteration

```
# CPI
ARIMA_cpi_2 <- arima(CPIGRL_ts, order=c(2,0,1),seasonal=list(order=c(1,0,1)), xreg = cbind(t))
coeftest(ARIMA_cpi_2)
```

```
##
## z test of coefficients:
##
##               Estimate Std. Error  z value              Pr(>|z|)
## ar1          1.2848759  0.0497062  25.8494 < 0.00000000000000022 ***
## ar2         -0.3052258  0.0442657  -6.8953     0.000000000005375 ***
## ma1         -0.8891884  0.0303200 -29.3268 < 0.00000000000000022 ***
## sar1         0.9685598  0.0138501  69.9316 < 0.00000000000000022 ***
## sma1        -0.8620535  0.0292447 -29.4772 < 0.00000000000000022 ***
## intercept  2.8622501 11.2494374   0.2544                0.7992
## t         -0.0013238  0.0056515  -0.2342                0.8148
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

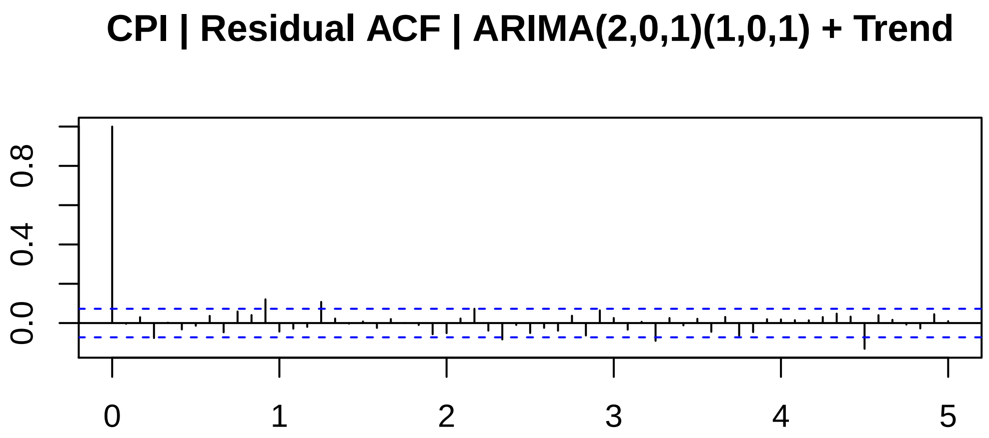
Both the added MA components are highly significant, but the trend and intercept became insignificant, which is not surprising if we look at the fitted trend plots above.

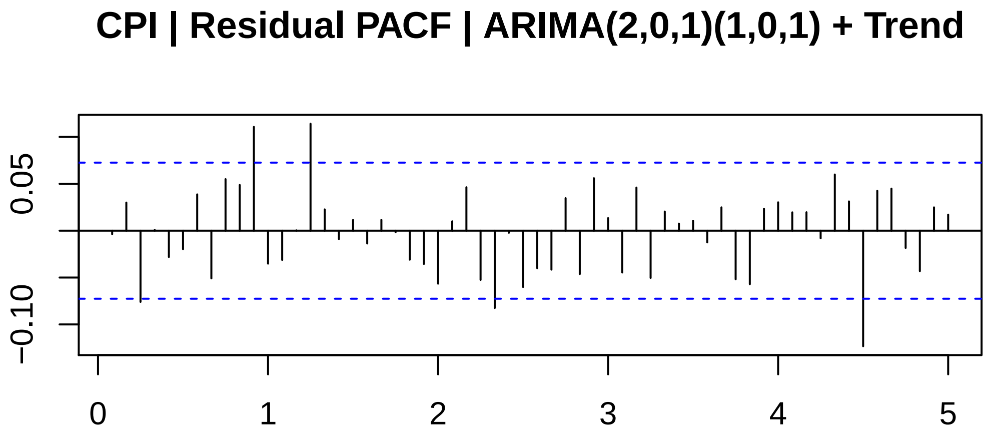
```
# UR
ARIMA_ur_2 <- arima(UNRATE_ts, order=c(1,1,1),seasonal=list(order=c(1,0,0)))
coeftest(ARIMA_ur_2)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error z value              Pr(>|z|)
## ar1  -0.899296   0.035767 -25.143 < 0.00000000000000022 ***
## ma1   0.954008   0.025113  37.988 < 0.00000000000000022 ***
## sar1  0.771268   0.038245  20.166 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

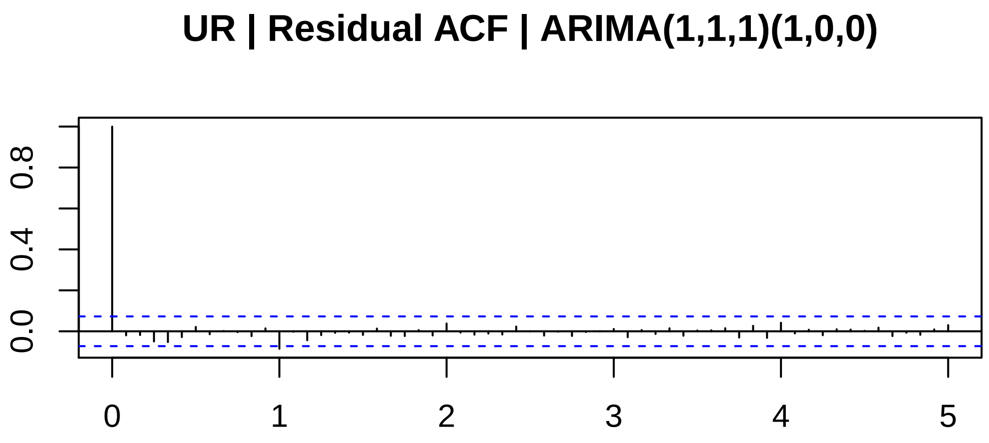
The added MA component is highly significant.

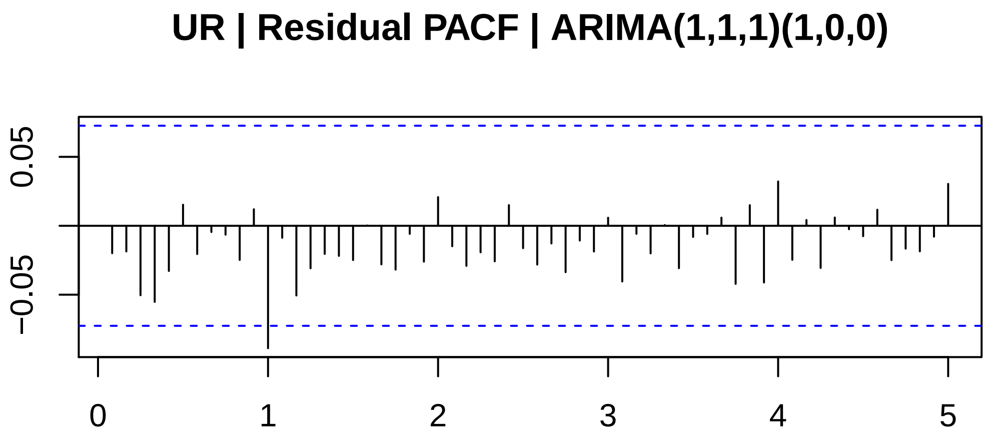
We again take a look at the ACF and PACF of the residuals to see if we can improve our model further.





We suggest increasing the MA lags and removing the intercept and trend in the next iteration.





Here, we also suggest increasing the MA lags.

ARIMA Models Third Iteration

```
# CPI

ARIMA_cpi_3.1 <-arima(CPIGRL_ts, order=c(2,0,2), seasonal=list(order=c(1,0,1)), include.mean=FAL
coeftest(ARIMA_cpi_3.1)
```

```
##
## z test of coefficients:
##
##        Estimate Std. Error  z value              Pr(>|z|)
## ar1  -0.155860   0.089007  -1.7511             0.0799285 .
## ar2   0.732918   0.061589  11.9001 < 0.00000000000000022 ***
## ma1   0.556966   0.119837   4.6477           0.000003357 ***
## ma2  -0.385892   0.100424  -3.8426             0.0001217 ***
## sar1  0.971301   0.010702  90.7571 < 0.00000000000000022 ***
## sma1 -0.831321   0.029511 -28.1701 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
ARIMA_cpi_3.2 <-arima(CPIGRL_ts, order=c(2,0,3), seasonal=list(order=c(1,0,1)), include.mean=FAL
coeftest(ARIMA_cpi_3.2)
```

```
##
## z test of coefficients:
##
##        Estimate Std. Error  z value              Pr(>|z|)
## ar1   0.641001   0.250151   2.5625              0.010394 *
## ar2   0.326352   0.242031   1.3484              0.177534
## ma1  -0.242413   0.246877  -0.9819              0.326139
## ma2  -0.346517   0.154650  -2.2407              0.025048 *
## ma3  -0.198345   0.059213  -3.3497              0.000809 ***
## sar1  0.972008   0.012327  78.8506 < 0.00000000000000022 ***
## sma1 -0.866743   0.027872 -31.0974 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
ARIMA_cpi_3.3 <-arima(CPIGRL_ts, order=c(2,0,4), seasonal=list(order=c(1,0,1)), include.mean=FAL
coeftest(ARIMA_cpi_3.3)
```

```
##
## z test of coefficients:
##
##        Estimate Std. Error  z value              Pr(>|z|)
## ar1   0.046021   0.054265   0.8481             0.3963980
## ar2   0.909267   0.051886  17.5243 < 0.00000000000000022 ***
## ma1   0.360789   0.064403   5.6021        0.000000021182 ***
## ma2  -0.684618   0.050370 -13.5917 < 0.00000000000000022 ***
## ma3  -0.254972   0.043116  -5.9137        0.000000003346 ***
## ma4  -0.129560   0.036325  -3.5667             0.0003615 ***
## sar1  0.970930   0.012471  77.8551 < 0.00000000000000022 ***
## sma1 -0.860924   0.028683 -30.0153 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We specify three ARIAM models with an increasing number of MA lags. We will use diagnostic statistic to evaluate which model is best.

```
# UR
ARIMA_ur_3 <- arima(UNRATE_ts, order=c(1,1,2),seasonal=list(order=c(1,0,0)))
coeftest(ARIMA_ur_3)
```

```
##
## z test of coefficients:
##
```

```
##       Estimate Std. Error  z value           Pr(>|z|)
## ar1  -0.888139   0.041038 -21.6421 <0.0000000000000002 ***
## ma1   0.920015   0.055720  16.5115 <0.0000000000000002 ***
## ma2  -0.029310   0.041904  -0.6994             0.4843
## sar1  0.770717   0.038328  20.1084 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The second MA component is not significant so we will likely keep the second model as our final model.

```
##                df      AIC
## ARIMA_cpi_auto 11 209.7514
## ARIMA_cpi_1     6 226.9000
## ARIMA_cpi_2     8 123.8989
## ARIMA_cpi_3.1   7 153.1541
## ARIMA_cpi_3.2   8 120.1938
## ARIMA_cpi_3.3   9 121.8320
```

```
##                df      BIC
## ARIMA_cpi_auto 11 260.2447
## ARIMA_cpi_1     6 254.4500
## ARIMA_cpi_2     8 160.6323
## ARIMA_cpi_3.1   7 185.2958
## ARIMA_cpi_3.2   8 156.9272
## ARIMA_cpi_3.3   9 163.1571
```

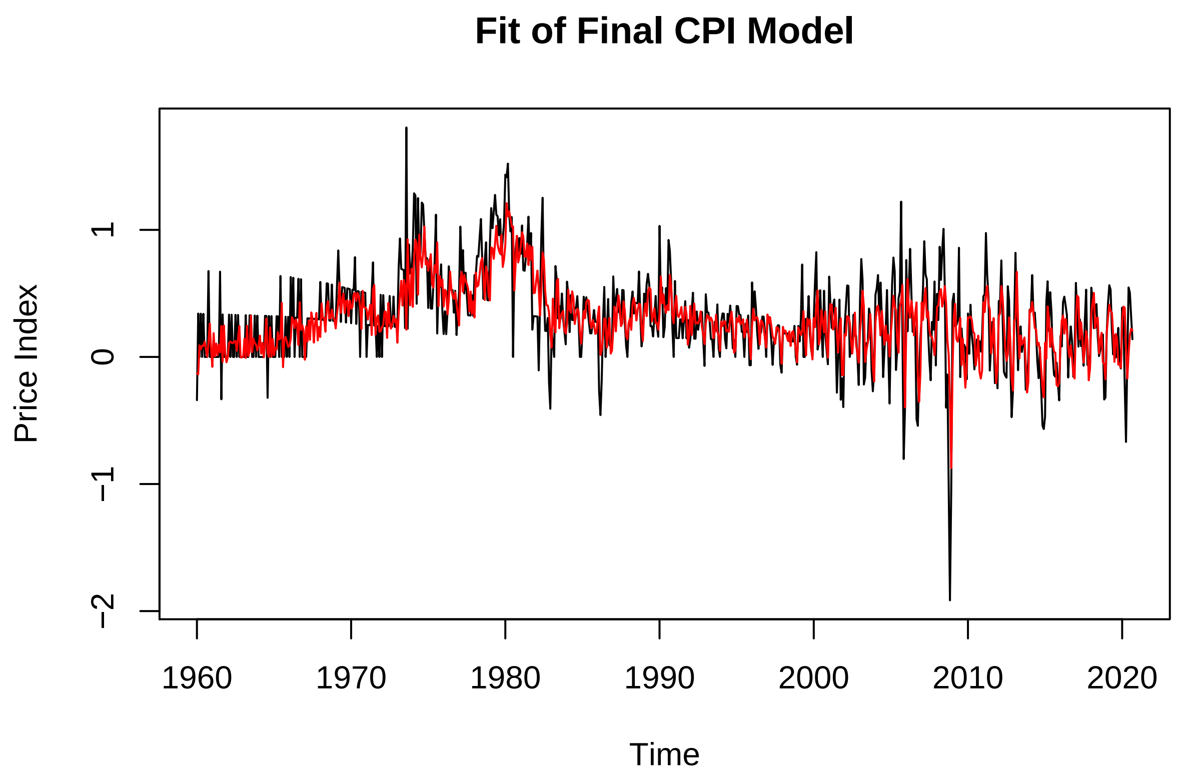
AIC and BIC agree that the model with three MA components is best. The information criteria of our proposed model are lower than for the model proposed by auto.arima. Our final model is ARIMA_cpi_3.2 [ARIMA(2,0,3)(1,0,1)]

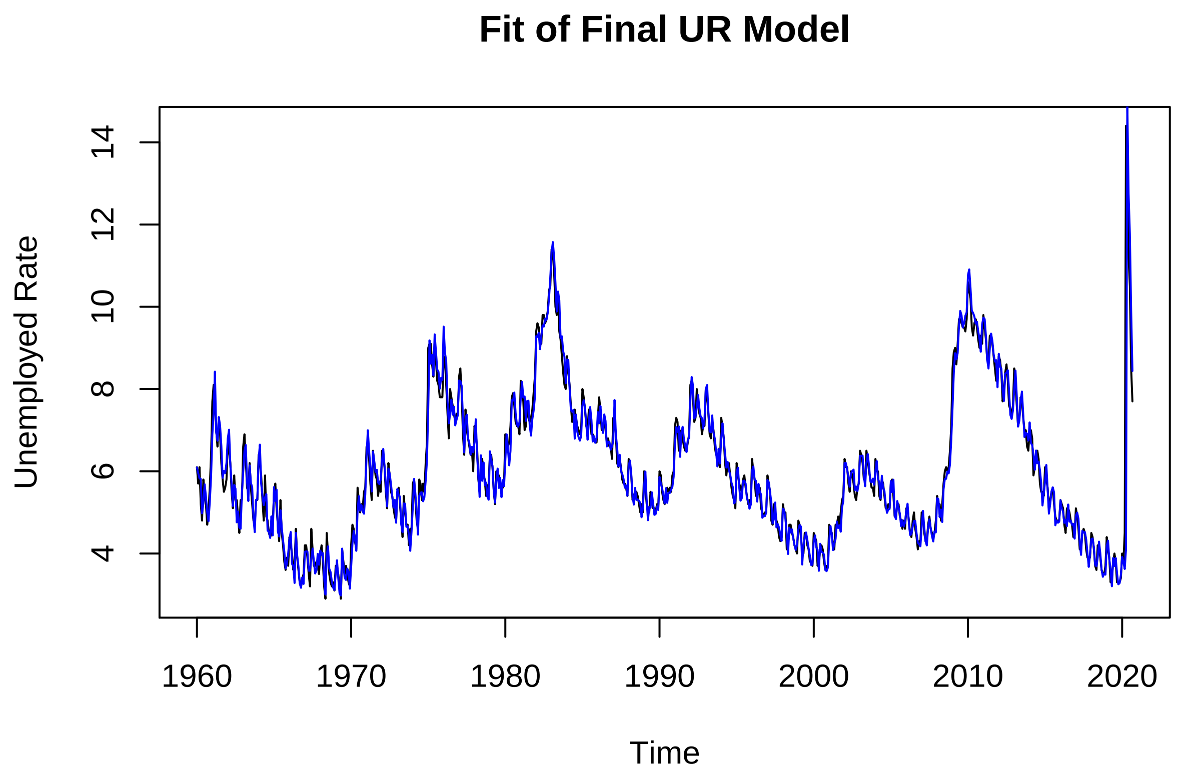
```
##               df       AIC
## ARIMA_ur_auto  4 1003.2230
## ARIMA_ur_1     3 1004.1388
## ARIMA_ur_2     4  997.4359
## ARIMA_ur_3     5  998.9449
```

```
##               df      BIC
## ARIMA_ur_auto  4 1021.584
## ARIMA_ur_1     3 1017.910
## ARIMA_ur_2     4 1015.797
## ARIMA_ur_3     5 1021.896
```

As expected, here both AIC and BIC agree that the second model is the best one. The information criteria of our proposed model are lower than for the model proposed by auto.arima. Our final model is ARIMA_ur_2 [ARIMA(1,1,1)(1,0,0)].

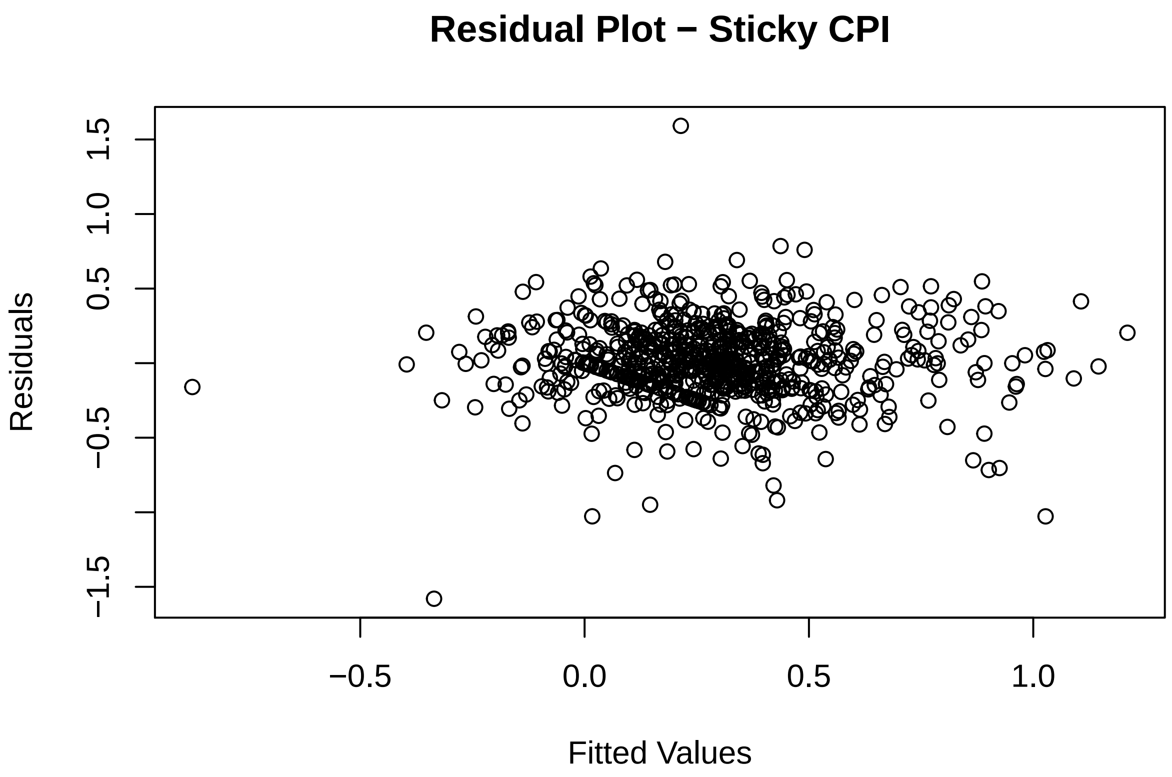
```
cpi_final_model <- arima(CPIGRL_ts, order=c(2,0,3), seasonal=list(order=c(1,0,1)), include.mean=
ur_final_model <- arima(UNRATE_ts, order=c(1,1,1),seasonal=list(order=c(1,0,0)))
```

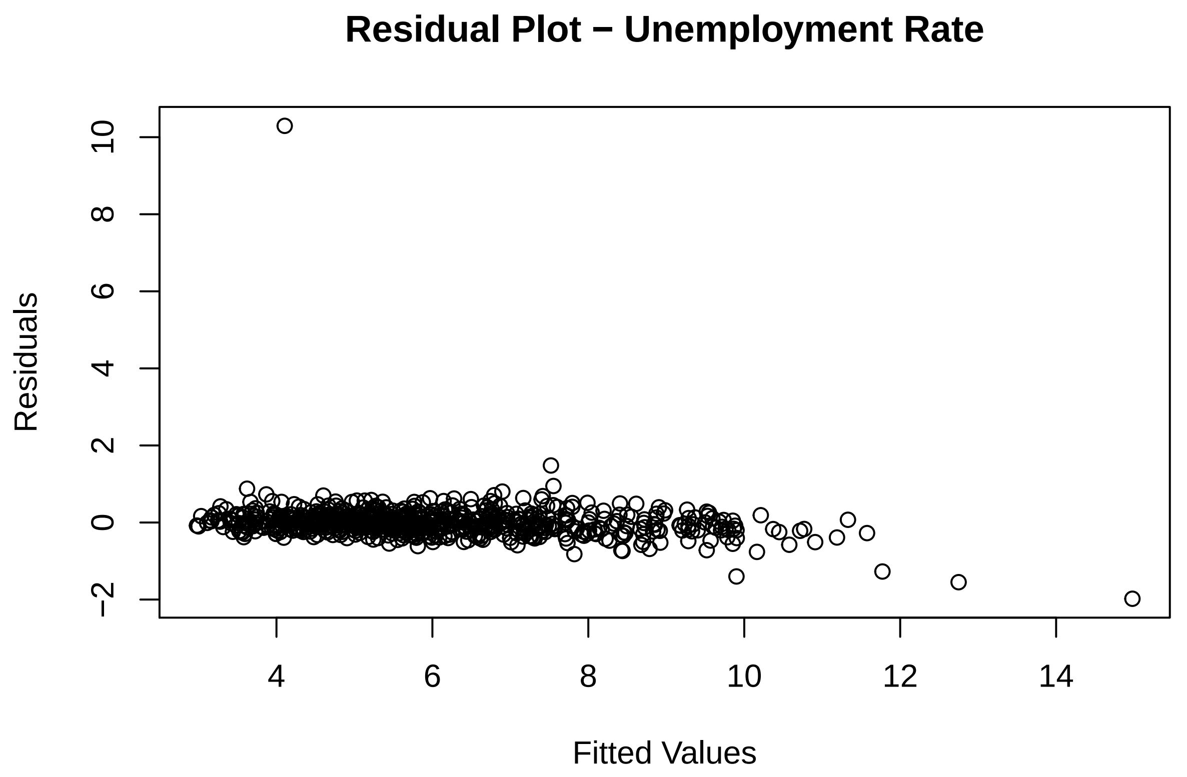




Residuals vs Fitted Values

Plot the respective residuals vs. fitted values and discuss your observations.


Residual Plot − Sticky CPI
Residuals
1.5
1.0
0.5
−0.5
−1.5
−0.5
0.0
0.5
1.0
Fitted Values

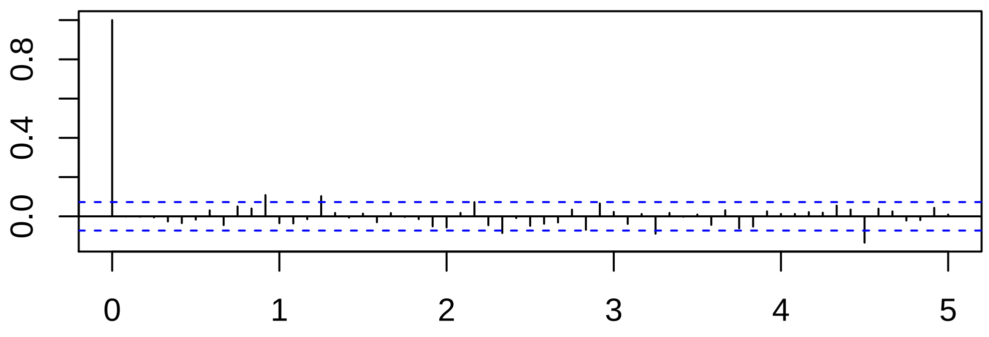


Trend mostly deviates around zero for both sticky CPI and unemployment rate, suggsting a linear relationship for both ARIMA models.

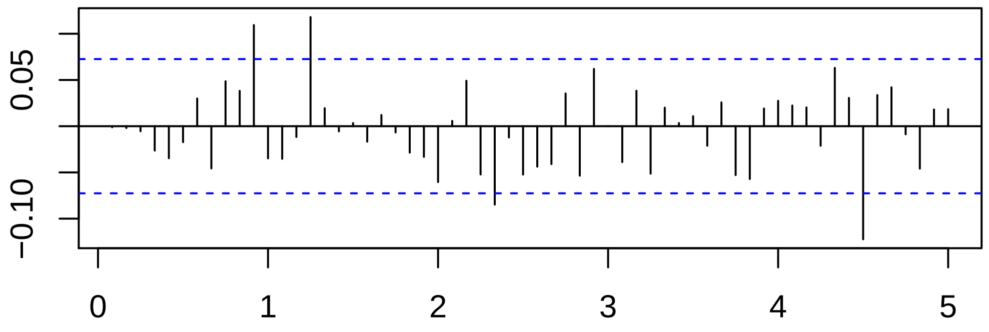
Residual ACF/PACF Plot

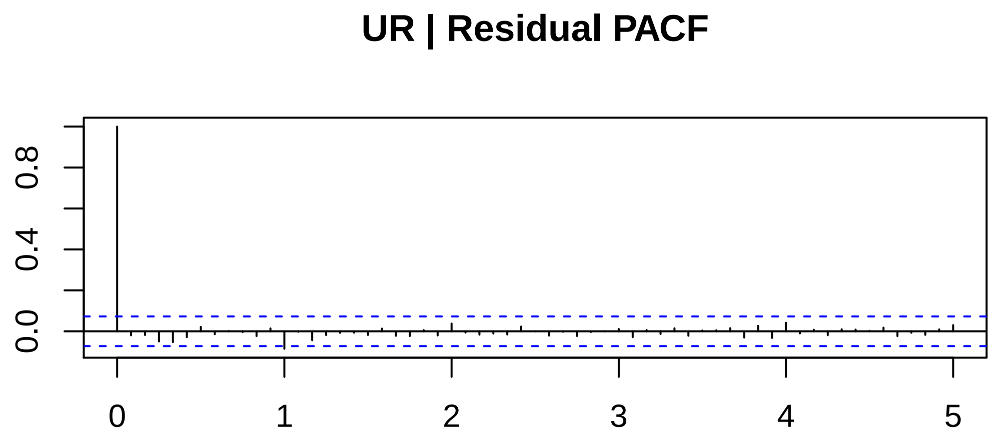
Plot the ACF and PACF of the respective residuals.

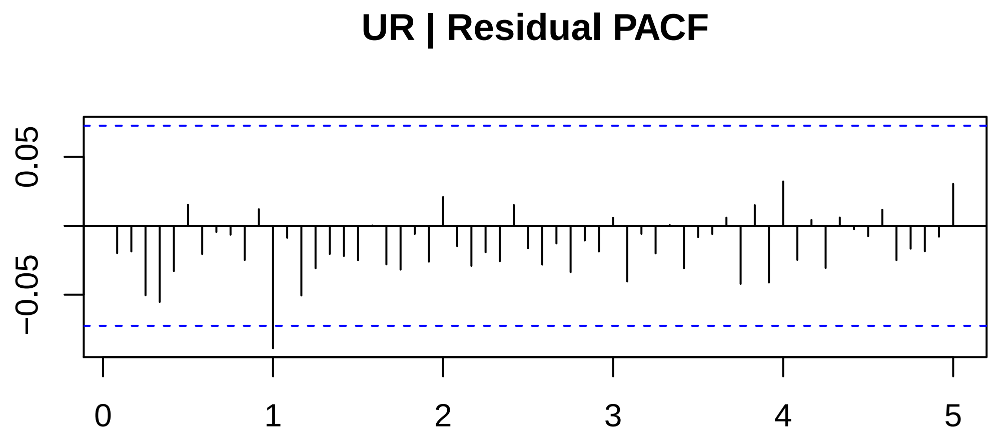
CPI | Residual ACF



CPI | Residual PACF

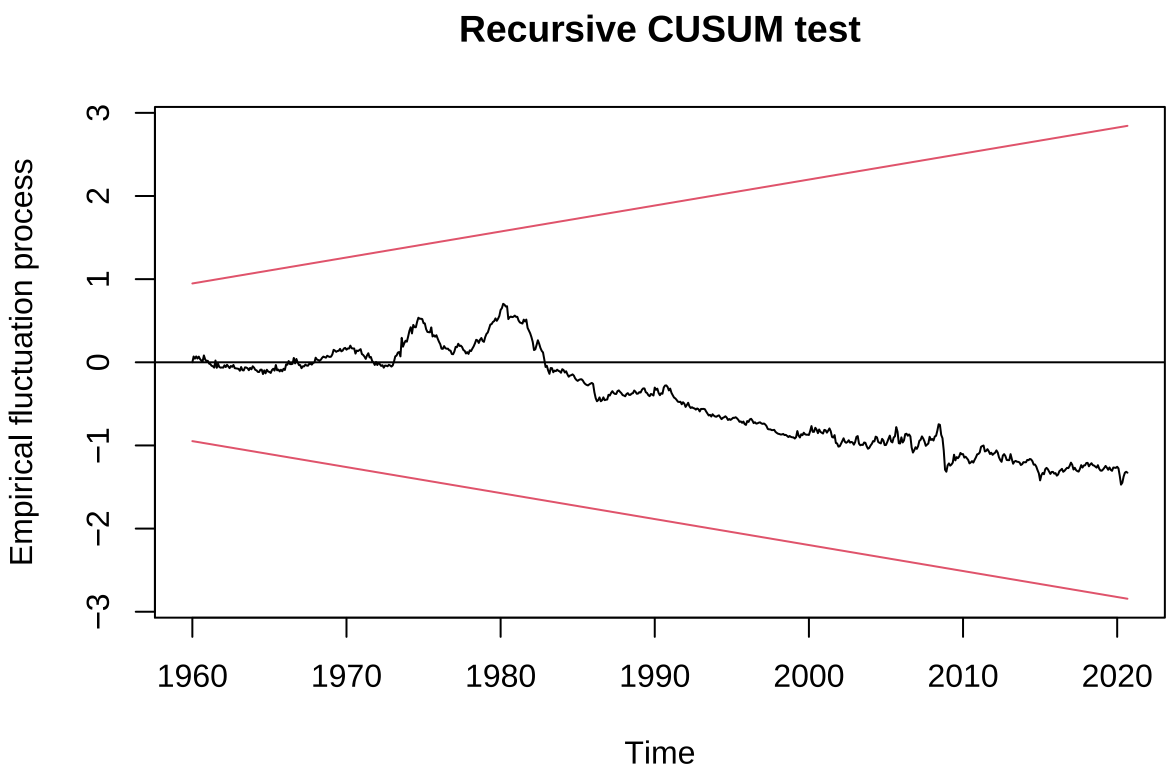


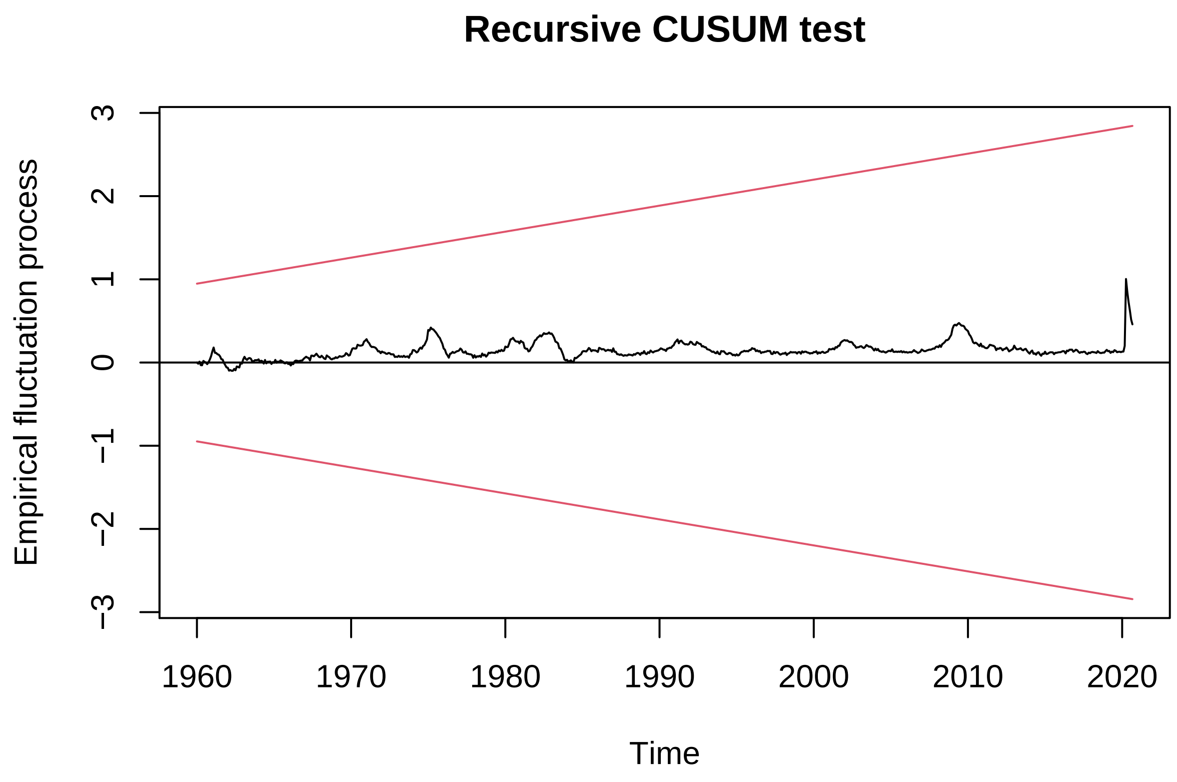




CUSUM Plot

Plot the respective CUSUM and interpret the plot.

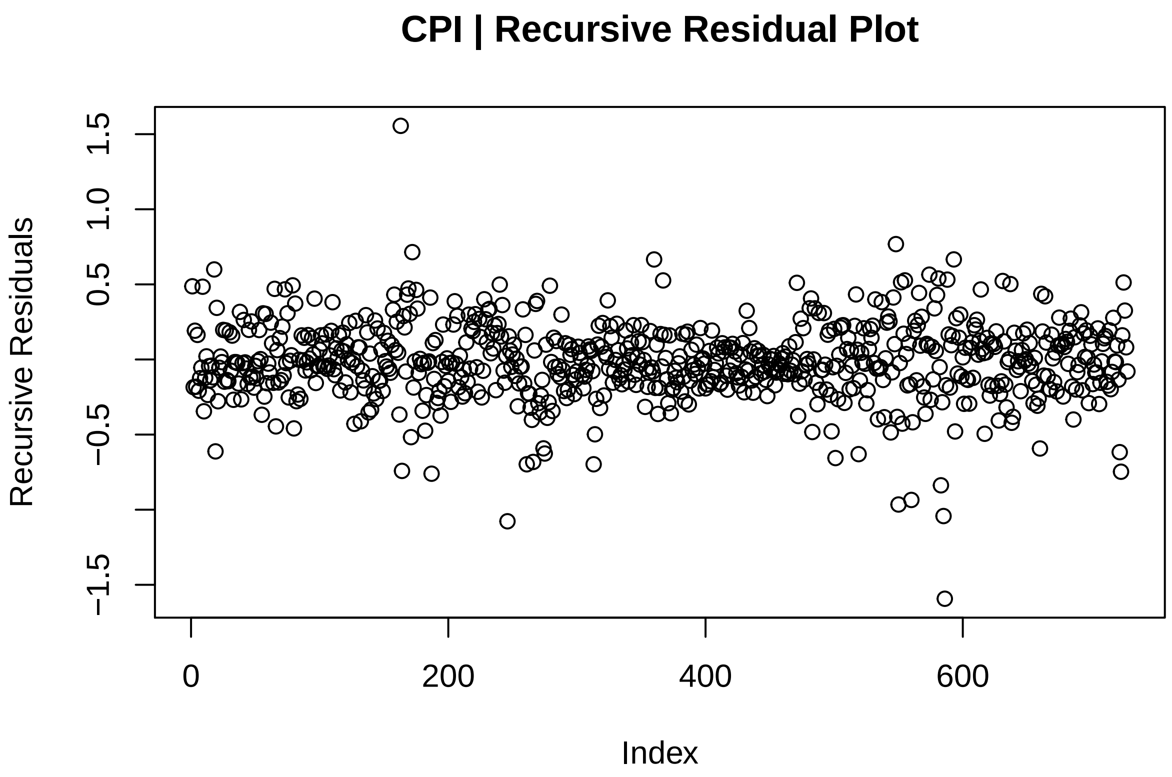

Recursive CUSUM test
Empirical fluctuation process
3
2
1
0
−1
−2
−3
1960
1970
1980
1990
2000
2010
2020
Time

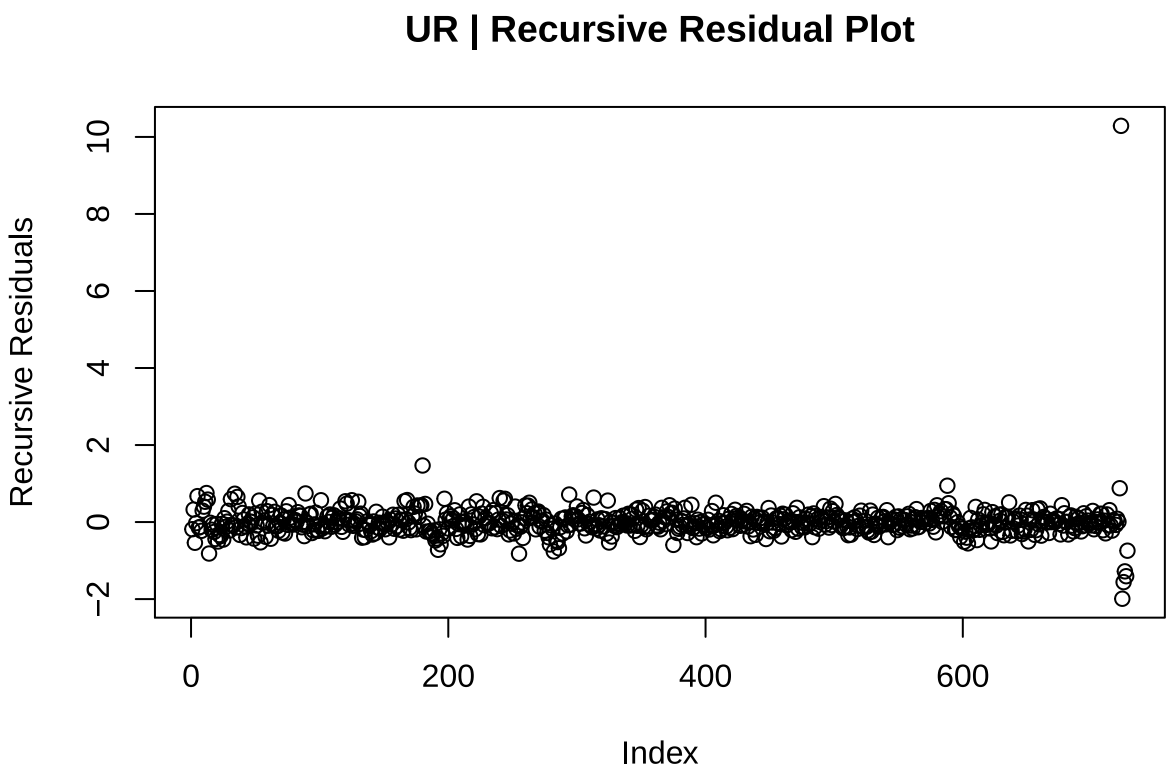


All points lie within the thresholds of the CUSUM test. This implies that the cumulative sum of deviations relative to the target value is "in control" or within tolerance.

Recursive Residuals Plot

Plot the respective Recursive Residuals and interpret the plot.


CPI | Recursive Residual Plot
Recursive Residuals
1.5
1.0
0.5
−0.5
−1.5
0
200
400
600
Index



Recursive residuals for both unemployment and CPI all mostly deviate around 0 with minimal outliers. This implies a pretty accurate model the fits the data really well, as seen in the actual vs fitted model in part 1(2).

Diagnostic Statistics (Ljung Box test, Box Pierce Test, RMSE)

For your model, discuss the associated diagnostic statistics.

We first take a look at the error measures of our models. We also include the error measures of the auto.arima models for reference.

```
##
## Call:
## arima(x = CPIGRL_ts, order = c(2, 0, 3), seasonal = list(order = c(1, 0, 1)),
##     include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2      ma1      ma2      ma3    sar1     sma1
##       0.6410  0.3264  -0.2424  -0.3465  -0.1983  0.9720  -0.8667
## s.e.  0.2502  0.2420   0.2469   0.1546   0.0592  0.0123   0.0279
##
## sigma^2 estimated as 0.06678:  log likelihood = -52.1,  aic = 120.19
##
```

```
## Training set error measures:
##                     ME      RMSE      MAE MPE MAPE      MASE        ACF1
## Training set 0.01146198 0.2584226 0.186938 NaN  Inf 0.7884151 -0.00115028
```

```
##
##  Accuracy of Auto.ARIMA Model:
```

```
##                      ME      RMSE       MAE MPE MAPE      MASE         ACF1
## Training set 0.001125553 0.2747209 0.2024569 NaN  Inf 0.7642513 0.0001256545
```

Our final CPI model has a lower Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) than the auto.arima model.

```
##
## Call:
## arima(x = UNRATE_ts, order = c(1, 1, 1), seasonal = list(order = c(1, 0, 0)))
##
## Coefficients:
##           ar1     ma1    sar1
##       -0.8993  0.9540  0.7713
## s.e.   0.0358  0.0251  0.0382
##
## sigma^2 estimated as 0.2247:  log likelihood = -494.72,  aic = 997.44
##
## Training set error measures:
##                      ME      RMSE       MAE          MPE     MAPE      MASE
## Training set 0.004037075 0.4737074 0.2153206 -0.002285853 3.481801 0.6089876
##                     ACF1
## Training set -0.01989685
```

```
##
##  Accuracy of Auto.ARIMA Model:
```

```
##                      ME     RMSE       MAE          MPE     MAPE      MASE
## Training set 0.004695828 0.475717 0.2170508 -0.006248699 3.516364 0.2644442
##                     ACF1
## Training set 0.001006954
```

Our final UR model has a slightly lower Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) than the auto.arima model.

We next test for autocorrelation in the two times series using the Ljung-Box and Box-Pierce test. Here, the null and alternative hypotheses are:

Ho : the model could be a white noise process H1 : the model might not be a white noise process

A significant p-value in the test rejects the null hypothesis that the time series is not autocorrelated. We reject the null if the p-value < 0.05

```
Box.test(recresid(cpi_final_model$res~1), type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  recresid(cpi_final_model$res ~ 1)
## X-squared = 0.0032634, df = 1, p-value = 0.9544
```

```
Box.test(recresid(cpi_final_model$res~1), type = "Box-Pierce")
```

```
##
##  Box-Pierce test
##
## data:  recresid(cpi_final_model$res ~ 1)
## X-squared = 0.00325, df = 1, p-value = 0.9545
```

Ljung-Box and Box-Pierce tests on the recursive residuals of the model have a p value of 0.9545, which is greater than all relevant significance levels. We fail to reject H0, which implies that there is no strong evidence for serial autocorrelation in the residuals of the model.

```
Box.test(recresid(ur_final_model$res~1),  type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  recresid(ur_final_model$res ~ 1)
## X-squared = 0.29383, df = 1, p-value = 0.5878
```

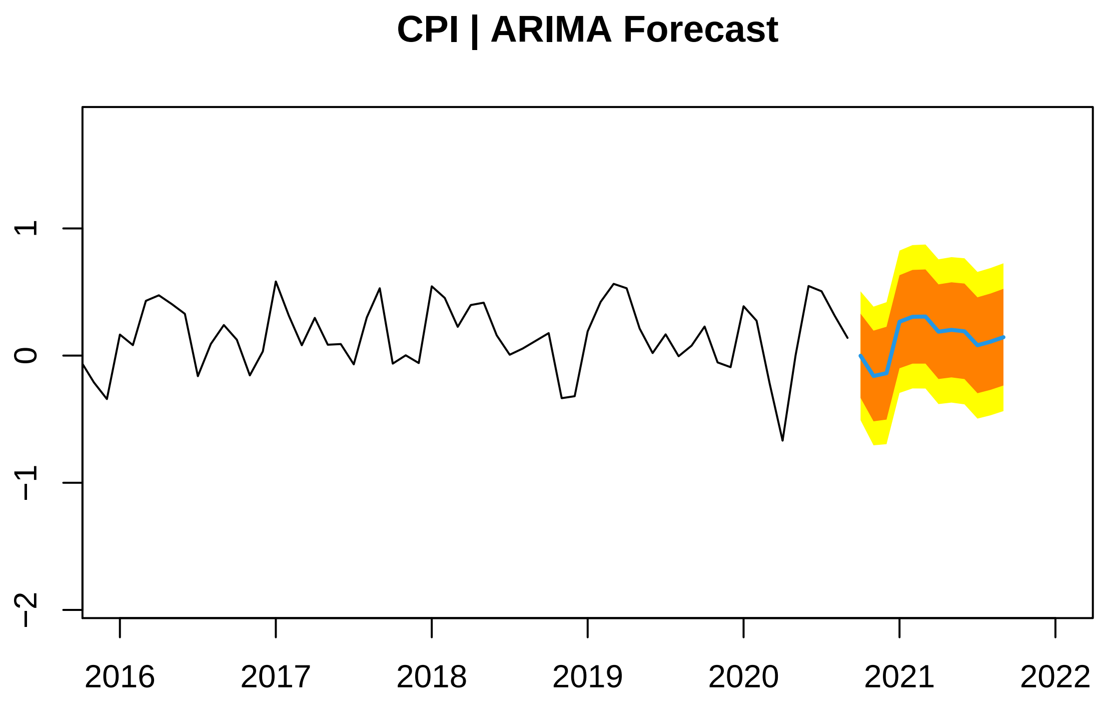
```
Box.test(recresid(ur_final_model$res~1), type = "Box-Pierce")
```

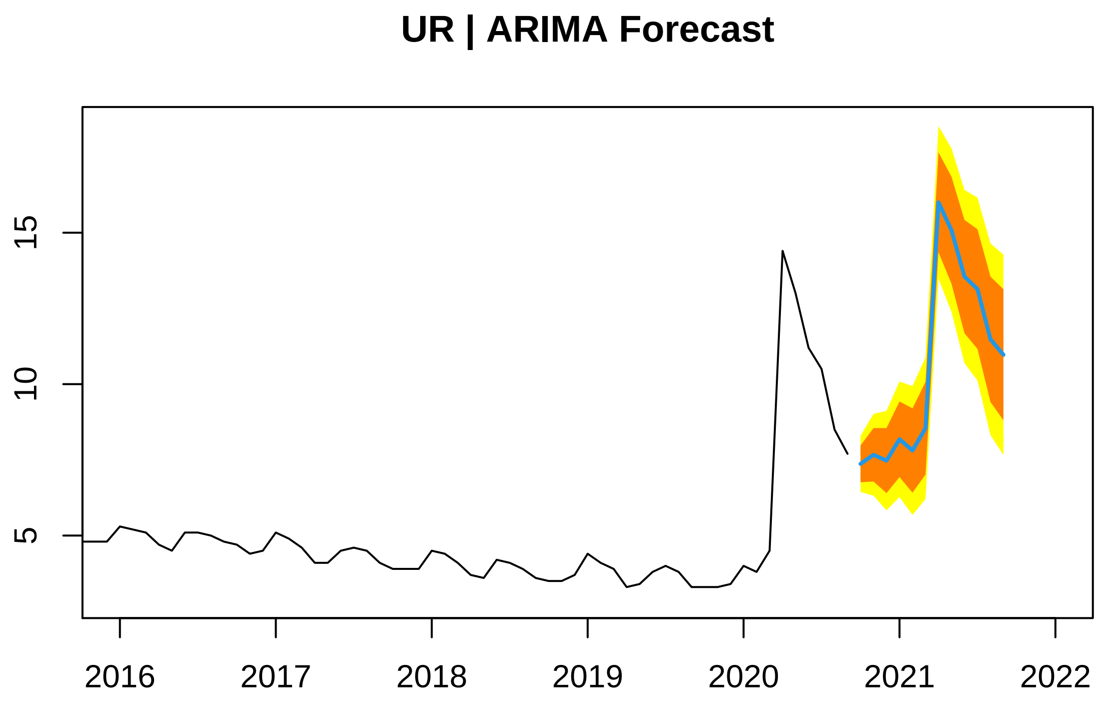
```
##
##  Box-Pierce test
##
## data:  recresid(ur_final_model$res ~ 1)
## X-squared = 0.29262, df = 1, p-value = 0.5885
```

We again fail to reject H0, so the Ljung-Box and Box-Pierce tests suggest that there is no strong evidence of serial autocorrelation in the residuals of the final UR model.

12 Step Forecast

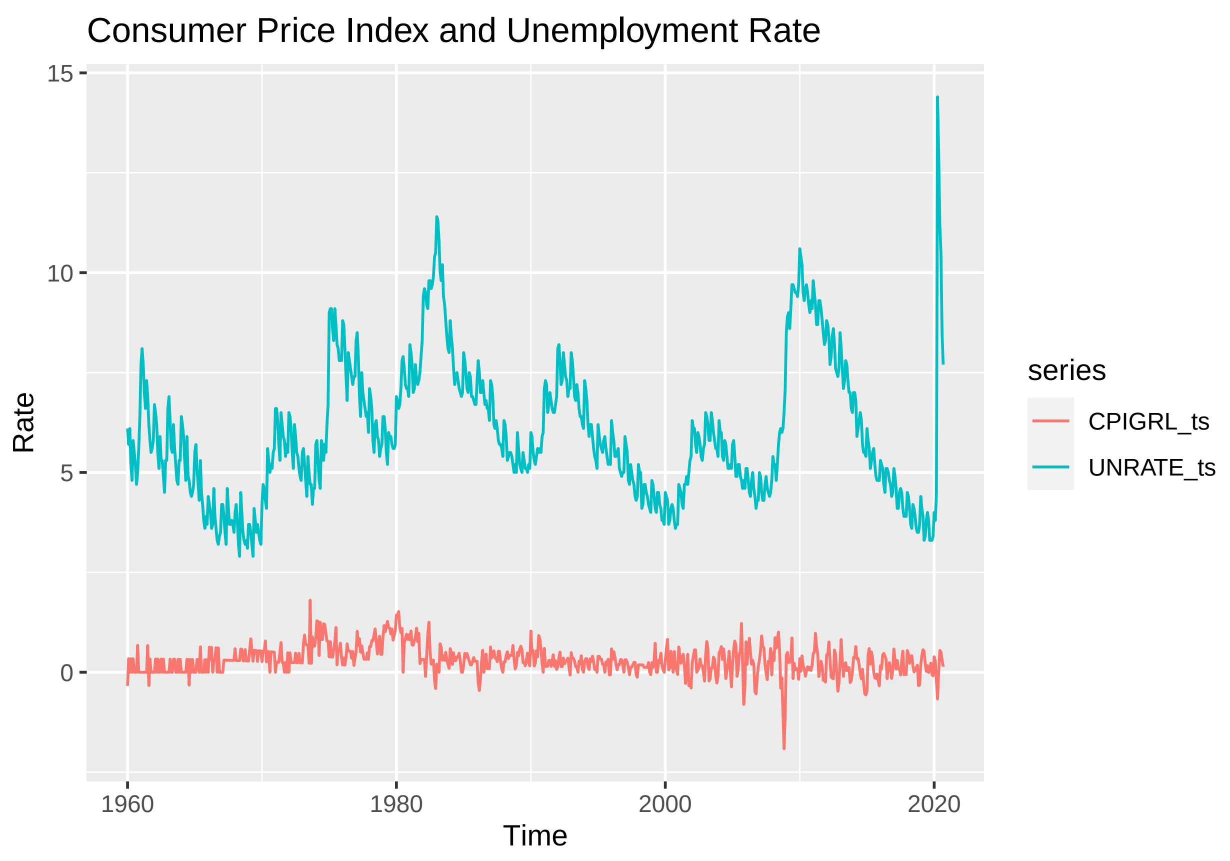
Use your model to forecast 12-steps ahead. Your forecast should include the respective error bands.


CPI | ARIMA Forecast
1
0
−1
−2
2016
2017
2018
2019
2020
2021
2022



Multivariate VAR Model

Fit an appropriate VAR model using your two variables. Make sure to show the relevant plots and discuss your results from the fit.



```
#Removing the seasonal component for both unemployment and CPI for VAR
desCPIGR<-de_CPIGRL_ts$x-de_CPIGRL_ts$seasonal
desUR<-de_UNRATE_ts$x-de_UNRATE_ts$seasonal
```

```
#y=cbind(CPIGRL_ts, UNRATE_ts)
y=cbind(desCPIGR, desUR)
lagselect<-VARselect(y,type="const")
lagselect$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     10      2      2     10
```

Lag chosen should be 10.

```
#detach("package:MTS", unload = TRUE)
y_model=VAR(y,p=10, type="const")
summary(y_model)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: desCPIGR, desUR
## Deterministic variables: const
## Sample size: 719
## Log Likelihood: -484.285
## Roots of the characteristic polynomial:
## 0.9573 0.9573 0.892 0.892 0.8775 0.8643 0.8643 0.8552 0.8552 0.8384 0.8384 0.8208 0.8208 0.82
## Call:
## VAR(y = y, p = 10, type = "const")
##
##
## Estimation results for equation desCPIGR:
## =========================================
## desCPIGR = desCPIGR.l1 + desUR.l1 + desCPIGR.l2 + desUR.l2 + desCPIGR.l3 + desUR.l3 + desCPIG
##
##               Estimate Std. Error t value             Pr(>|t|)
## desCPIGR.l1   0.437303   0.037805  11.567 < 0.0000000000000002 ***
## desUR.l1      0.003984   0.021548   0.185              0.85337
## desCPIGR.l2   0.065878   0.040961   1.608              0.10822
## desUR.l2     -0.012686   0.030027  -0.422              0.67280
## desCPIGR.l3  -0.048685   0.041272  -1.180              0.23855
## desUR.l3     -0.014828   0.030253  -0.490              0.62420
## desCPIGR.l4   0.084582   0.041060   2.060              0.03977 *
## desUR.l4      0.020588   0.030427   0.677              0.49886
## desCPIGR.l5   0.012594   0.041319   0.305              0.76061
## desUR.l5     -0.023841   0.030439  -0.783              0.43376
## desCPIGR.l6   0.039745   0.041340   0.961              0.33667
## desUR.l6      0.092002   0.051332   1.792              0.07352 .
## desCPIGR.l7   0.081733   0.041057   1.991              0.04690 *
## desUR.l7     -0.152612   0.059069  -2.584              0.00998 **
## desCPIGR.l8  -0.029760   0.041038  -0.725              0.46858
## desUR.l8      0.162792   0.060187   2.705              0.00700 **
## desCPIGR.l9   0.096716   0.040739   2.374              0.01786 *
## desUR.l9     -0.061598   0.059194  -1.041              0.29842
## desCPIGR.l10  0.105855   0.038128   2.776              0.00565 **
## desUR.l10    -0.016168   0.043576  -0.371              0.71072
## const         0.060848   0.040220   1.513              0.13076
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.26 on 698 degrees of freedom
## Multiple R-Squared: 0.4186,  Adjusted R-squared: 0.4019
## F-statistic: 25.13 on 20 and 698 DF,  p-value: < 0.00000000000000022
##
##
## Estimation results for equation desUR:
## ======================================
## desUR = desCPIGR.l1 + desUR.l1 + desCPIGR.l2 + desUR.l2 + desCPIGR.l3 + desUR.l3 + desCPIGR.l
```

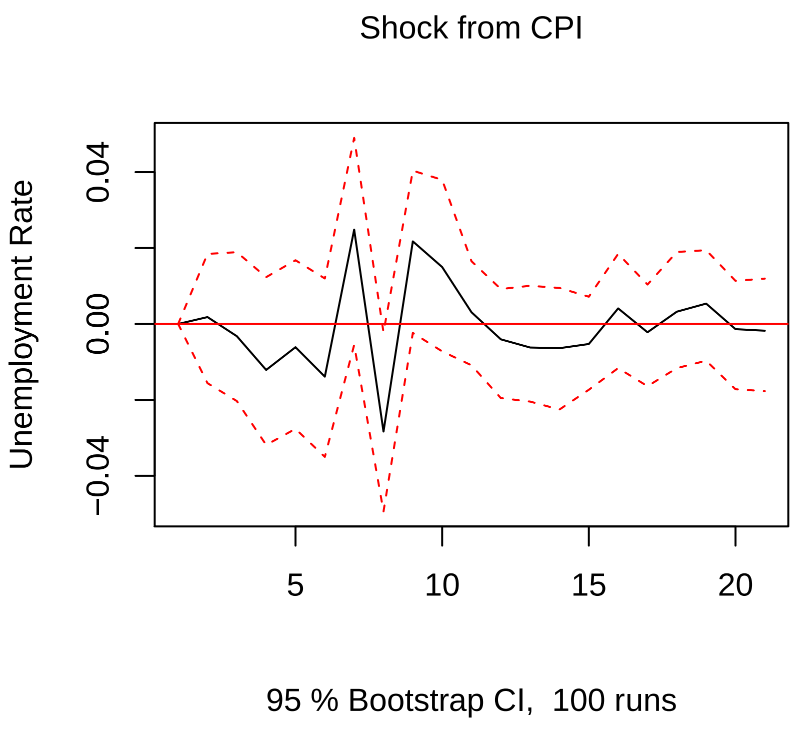
```
##
##               Estimate Std. Error t value             Pr(>|t|)
## desCPIGR.l1  -0.140155   0.066941  -2.094              0.03665 *
## desUR.l1      0.968175   0.038155  25.375 < 0.0000000000000002 ***
## desCPIGR.l2   0.071330   0.072529   0.983              0.32572
## desUR.l2     -0.105288   0.053169  -1.980              0.04807 *
## desCPIGR.l3   0.051627   0.073079   0.706              0.48014
## desUR.l3      0.088703   0.053568   1.656              0.09819 .
## desCPIGR.l4   0.101873   0.072705   1.401              0.16160
## desUR.l4     -0.082784   0.053876  -1.537              0.12485
## desCPIGR.l5  -0.070807   0.073163  -0.968              0.33348
## desUR.l5      0.066742   0.053898   1.238              0.21602
## desCPIGR.l6   0.165315   0.073200   2.258              0.02423 *
## desUR.l6      0.057818   0.090892   0.636              0.52490
## desCPIGR.l7  -0.024370   0.072700  -0.335              0.73756
## desUR.l7      0.005108   0.104592   0.049              0.96107
## desCPIGR.l8   0.014856   0.072665   0.204              0.83807
## desUR.l8      0.064013   0.106573   0.601              0.54827
## desCPIGR.l9   0.019132   0.072135   0.265              0.79091
## desUR.l9      0.107198   0.104814   1.023              0.30679
## desCPIGR.l10 -0.054950   0.067513  -0.814              0.41597
## desUR.l10    -0.206880   0.077159  -2.681              0.00751 **
## const         0.189536   0.071216   2.661              0.00796 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.4604 on 698 degrees of freedom
## Multiple R-Squared: 0.9284,  Adjusted R-squared: 0.9264
## F-statistic: 452.7 on 20 and 698 DF,  p-value: < 0.00000000000000022
##
##
##
## Covariance matrix of residuals:
##          desCPIGR    desUR
## desCPIGR  0.06761 -0.01844
## desUR    -0.01844  0.21197
##
## Correlation matrix of residuals:
##          desCPIGR  desUR
## desCPIGR    1.000 -0.154
## desUR      -0.154  1.000
```

There seems to be a negative relationship between the CPI growth rate and the unemployment rate at different lags for both CPI and unemployment. For example, in lag 1, both CPI growth rate and unemployment are significant variables that affect the unemployment rate, though the effects of CPI growth rate become much less significant as the number of lags increase.

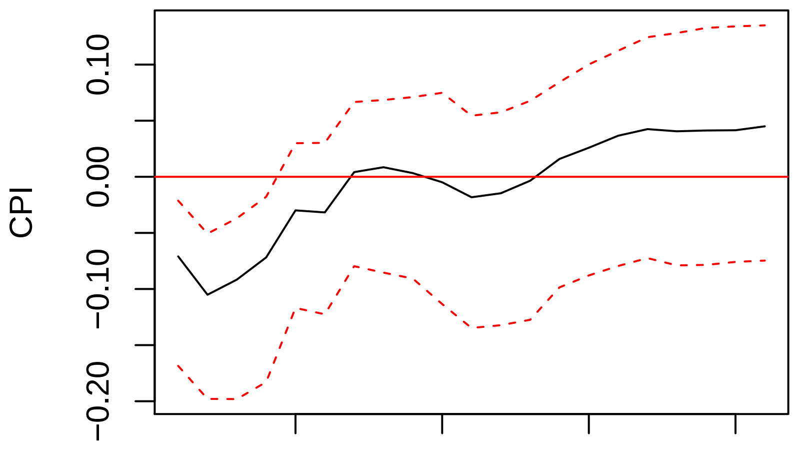
The unemployment rate in lag 7 and 8 on the other hand seems to significantly affect the CPI growth rate, but remain mostly insignificant for the rest of the lags.

Impulse Response Function Plot

Compute, plot, and interpret the respective impulse response functions.



Shock from Unemployment Rate



95 % Bootstrap CI, 100 runs

For unemployment rate, there is an unclear relationship based on how it responds to shocks from CPI. While the pattern deviates around zero, the pattern is mostly irregular and hard to predict. Further tests are needed to see whether an actual relationship between the two can be determined, which is done in part 12.

There seems to be a positive relationship between how CPI responds to a shock from the unemployment rate, though the respones mostly deviates around zero. Further tests are needed to ensure that there is a clear relationship, which will be done in part 12.

Granger Causality Test

Perform a Granger-Causality test on your variables and discuss your results from the test.

```
##
##  Granger causality H0: desCPIGR do not Granger-cause desUR
##
## data:  VAR object y_model
## F-Test = 1.9469, df1 = 10, df2 = 1396, p-value = 0.03558
```

```
##
##  Granger causality H0: desUR do not Granger-cause desCPIGR
##
## data:  VAR object y_model
## F-Test = 1.5487, df1 = 10, df2 = 1396, p-value = 0.1166
```

CPI growth rate Granger causes unemployment rate at the 5% significant level, but unemployment rate does not Granger cause CPI growth. This is interesting, as the short run Philip's Curve predicts that inflation rate and unemployment rate both have a causal relationship with each other. Our results, however suggests that the unemployment rate is more responsive to changes in CPI growth rate, while the CPI growth rate does not necessarily revolve around changes in the unemployment rate.

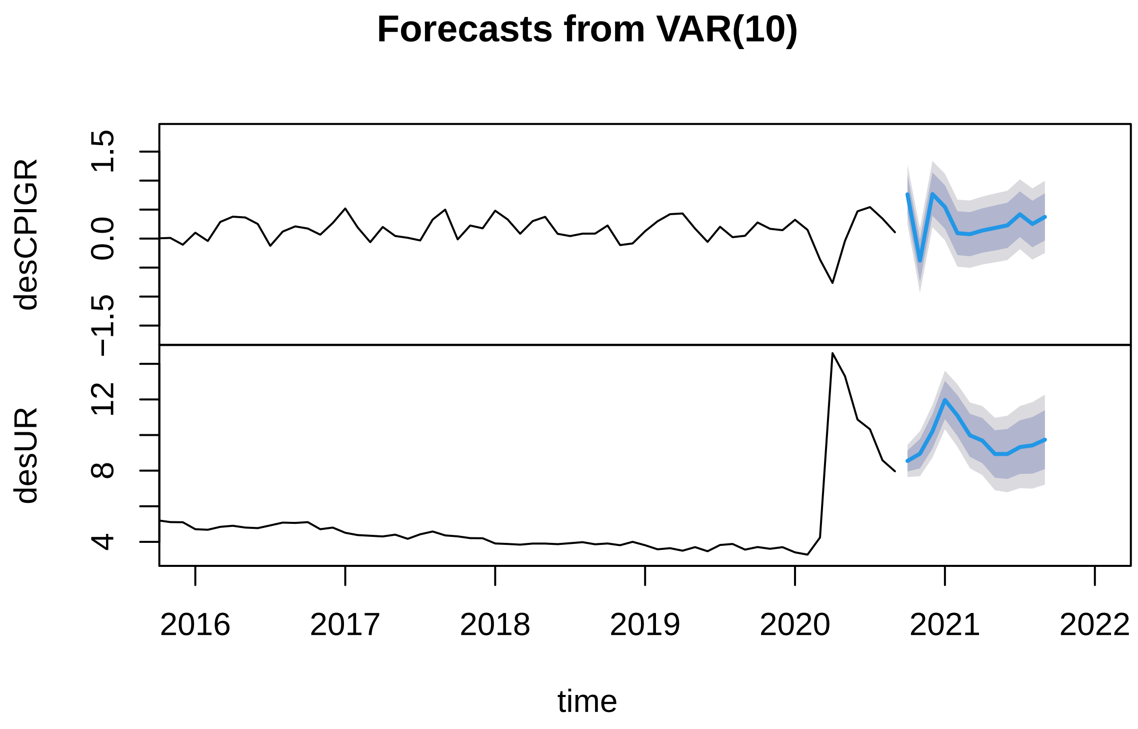
VAR Model Forecast

Use your VAR model to forecast 12-steps ahead. Your forecast should include the respective error bands. Comment on the differences between the two forecasts (VAR vs. ARIMA).

```
forecast(y_model, h=12, level=c(95))
```

```
## desCPIGR
##          Point Forecast       Lo 95     Hi 95
## Oct 2020     0.76310177  0.25348311 1.2727204
## Nov 2020    -0.37956491 -0.93557108 0.1764413
## Dec 2020     0.76763016  0.19619733 1.3390630
## Jan 2021     0.54515186 -0.02907909 1.1193828
## Feb 2021     0.09310886 -0.48495300 0.6711707
## Mar 2021     0.07580976 -0.50541183 0.6570314
## Apr 2021     0.13970879 -0.44574561 0.7251632
## May 2021     0.18355233 -0.41055246 0.7776571
## Jun 2021     0.22817899 -0.36933201 0.8256900
## Jul 2021     0.41938163 -0.18340569 1.0221690
## Aug 2021     0.25168797 -0.36251681 0.8658928
## Sep 2021     0.37358506 -0.24856210 0.9957322
##
## desUR
##          Point Forecast     Lo 95     Hi 95
## Oct 2020       8.545601  7.643230  9.447973
## Nov 2020       8.951096  7.685446 10.216747
## Dec 2020      10.210575  8.732814 11.688337
## Jan 2021      11.970179 10.325904 13.614454
## Feb 2021      11.092378  9.336790 12.847967
## Mar 2021       9.983452  8.135896 11.831007
## Apr 2021       9.685537  7.742690 11.628383
## May 2021       8.933928  6.897348 10.970507
## Jun 2021       8.936520  6.786665 11.086375
## Jul 2021       9.323818  7.018691 11.628945
## Aug 2021       9.420548  6.989526 11.851570
## Sep 2021       9.740128  7.215163 12.265092
```

```r
plot(forecast(y_model, h=12), xlim=c(2016,2022))
```



The forecast here seems flatter/less extreme than the forecast made in the ARIMA model. Note that this forecast omits seasonality, and hence is predicting the trend of the seasonally adjusted forecast.

ARIMA Model Backtest

Backtest your ARIMA model. Begin by partitioning your data set into an estimation set and a prediction set.

We decided to split the data into an estimation and prediciton set of roughly 50% each.

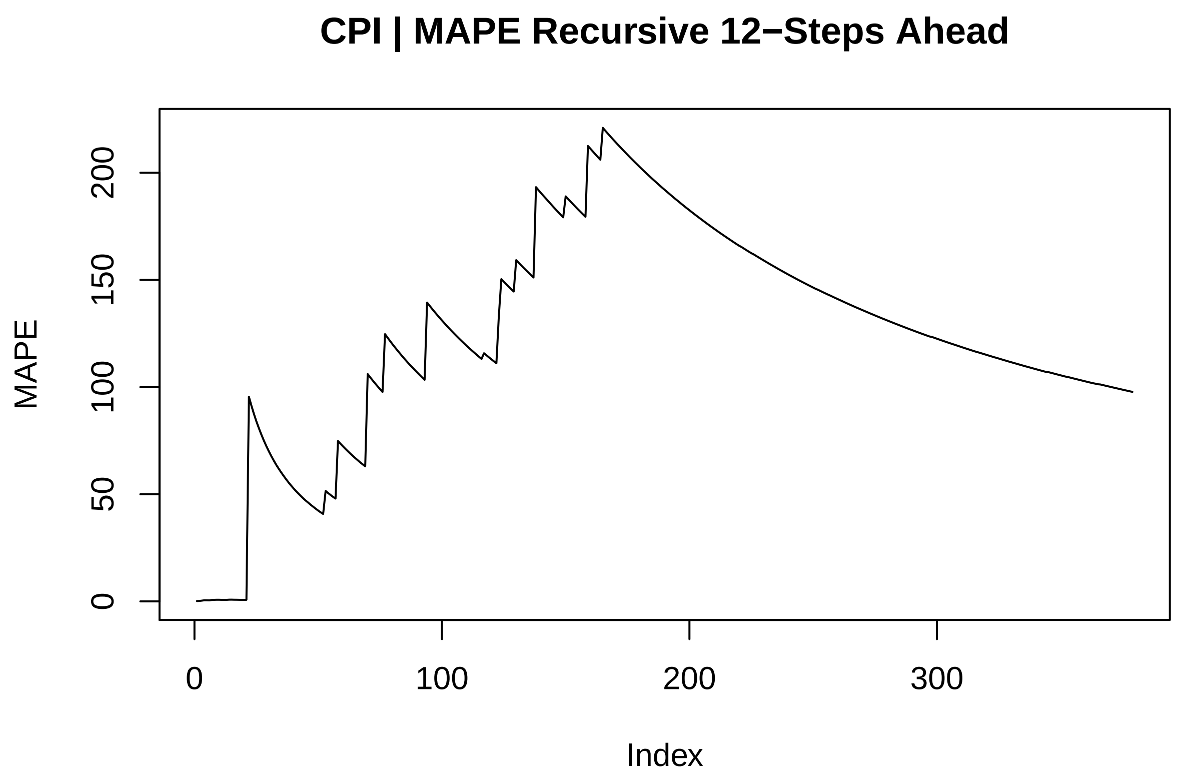
###Use a recursive backtesting scheme, and forecast 12-steps ahead at each iteration. Compute the mean absolute percentage error at each step. Provide a plot showing the MAPE over each iteration.

```r
# CPI 12-Steps ahead

#install.packages("MTS")
library(MTS)

CPIGRL_ts_mod <- CPIGRL_ts+0.0001
```

```r
recursive_cpi_12 <- MTS::backtest(cpi_final_model, CPIGRL_ts_mod, orig=350, h=12)
```

```
## [1] "RMSE of out-of-sample forecasts"
##  [1] 0.2972925 0.3275198 0.3024534 0.3239643 0.3014392 0.3192777 0.2977021
##  [8] 0.3166543 0.2912400 0.3145553 0.2946432 0.3186796
## [1] "Mean absolute error of out-of-sample forecasts"
##  [1] 0.2087188 0.2407120 0.2049472 0.2335543 0.2031877 0.2291971 0.2047337
##  [8] 0.2314081 0.2001067 0.2294545 0.2017675 0.2325933
```

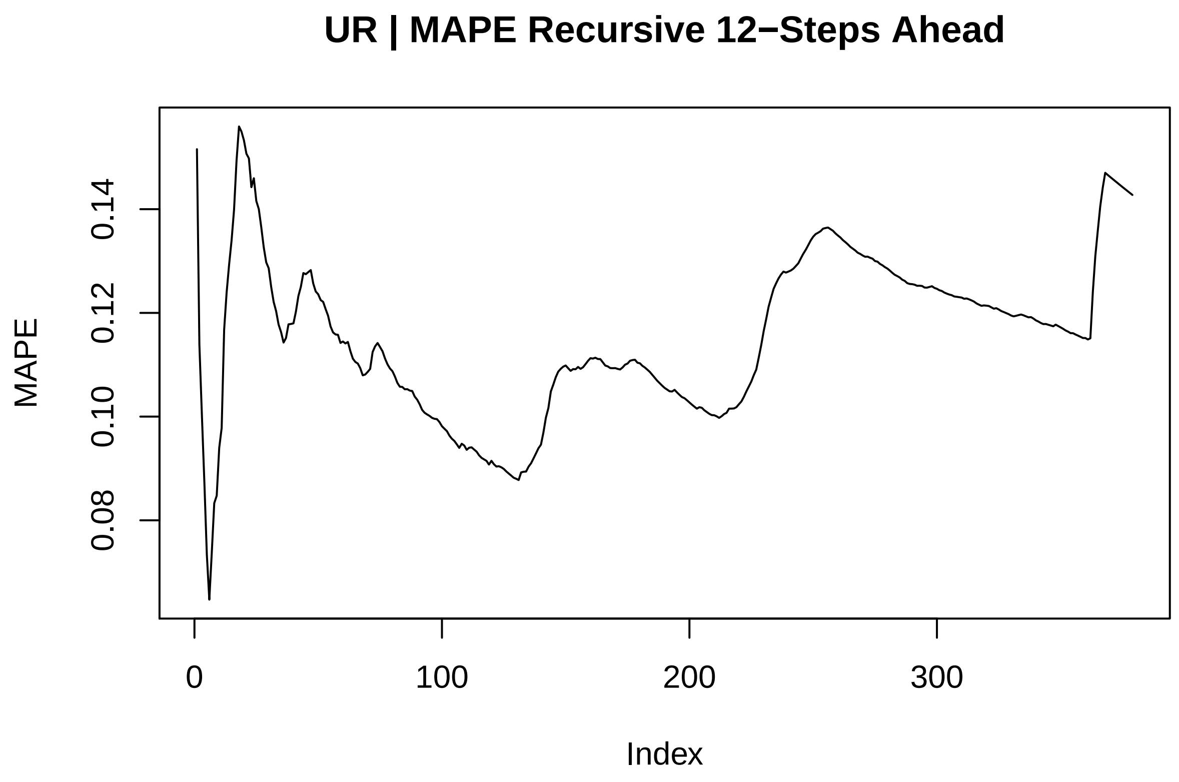
```r
cpi_error_12 <- data.frame(recursive_cpi_12$error) # Retrieve the errors from the backtest
cpi_error_12$actual <- CPIGRL_ts_mod[351:729] # Retrieve the actual data values from the data
cpi_error_12$abs <- abs(cpi_error_12$X12/cpi_error_12$actual) # calculate absolute error
cpi_error_12$mape <- cummean(cpi_error_12$abs) # calculate MAPE

plot(cpi_error_12$mape, type='l', main= "CPI | MAPE Recursive 12-Steps Ahead", ylab="MAPE")
```



```r
# UR 12-Steps ahead

recursive_ur_12 <- MTS::backtest(ur_final_model, UNRATE_ts, orig=350, h=12)
```

```
## [1] "RMSE of out-of-sample forecasts"
##  [1] 0.7531121 1.0844936 1.2709301 1.4609844 1.3371111 1.1900647 1.2045066
##  [8] 1.2598260 1.2760842 1.3333449 1.3584610 1.4222930
## [1] "Mean absolute error of out-of-sample forecasts"
##  [1] 0.3010883 0.4035631 0.4403244 0.5191229 0.5280725 0.5520116 0.5887671
##  [8] 0.6381557 0.6565491 0.6970864 0.7361395 0.7915500
```

```
ur_error_12 <- data.frame(recursive_ur_12$error)
ur_error_12$actual <- UNRATE_ts[351:729]
ur_error_12$abs <- abs(ur_error_12$X12/ur_error_12$actual)
ur_error_12$mape <- cummean(ur_error_12$abs)

plot(ur_error_12$mape, type='l', main= "UR | MAPE Recursive 12-Steps Ahead", ylab="MAPE")
```

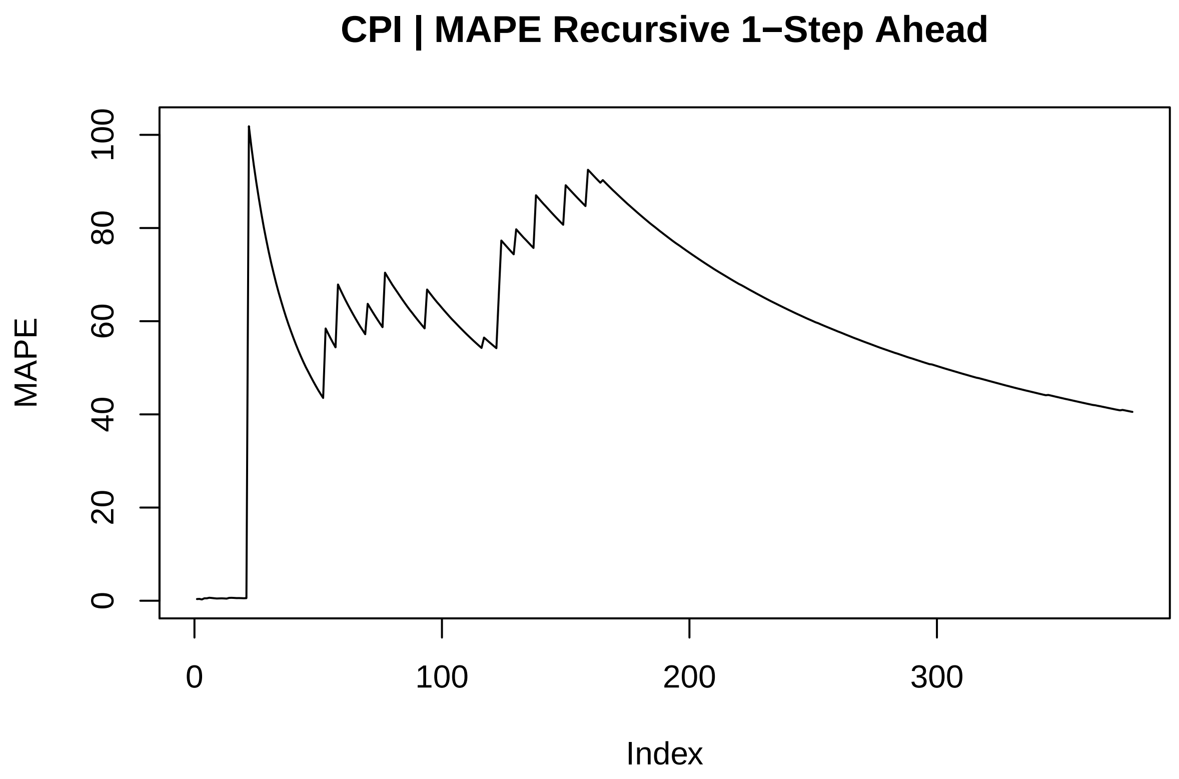


Shorten your forecast horizon to only 1-step ahead. Compute the absolute percentage error at each iteration, and plot.

```
# CPI 1-Step ahead

recursive_cpi_1 <- MTS::backtest(cpi_final_model, CPIGRL_ts_mod, orig=350, h=1)
```

```
## [1] "RMSE of out-of-sample forecasts"
## [1] 0.2972925
## [1] "Mean absolute error of out-of-sample forecasts"
## [1] 0.2087188
```

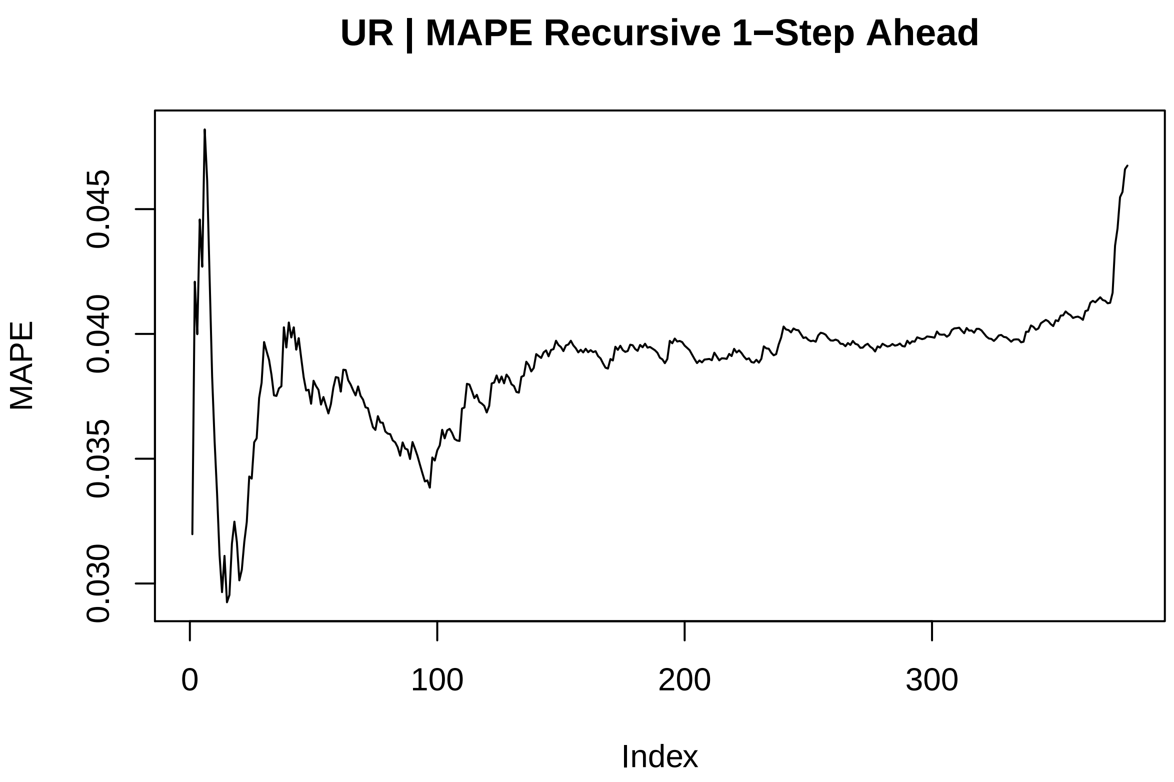
```
cpi_error_1 <- data.frame(recursive_cpi_1$error)
cpi_error_1$actual <- CPIGRL_ts_mod[351:729]+0.0001
cpi_error_1$abs <- abs(cpi_error_1$recursive_cpi_1.error/cpi_error_1$actual)
cpi_error_1$mape <- cummean(cpi_error_1$abs)

plot(cpi_error_1$mape, type='l', main= "CPI | MAPE Recursive 1-Step Ahead", ylab="MAPE")
```



```
# UR 1-Step ahead

recursive_ur_1 <- MTS::backtest(ur_final_model, UNRATE_ts, orig=350, h=1)
```

```
## [1] "RMSE of out-of-sample forecasts"
## [1] 0.7531121
## [1] "Mean absolute error of out-of-sample forecasts"
## [1] 0.3010883
```

```
ur_error_1 <- data.frame(recursive_ur_1$error)
ur_error_1$actual <- UNRATE_ts[351:729]
ur_error_1$abs <- abs(ur_error_1$recursive_ur_1.error/ur_error_1$actual)
ur_error_1$mape <- cummean(ur_error_1$abs)

plot(ur_error_1$mape, type='l', main= "UR | MAPE Recursive 1-Step Ahead", ylab="MAPE")
```



Based on your findings above, does your model perform better at longer or shorter horizon forecasts?

Looking at the scale of the y-axis alone, it seems that our models perform better with shorter horizon forecasts. We can confirm this by looking at the average MAPE which equals `mean(cpi_error_12$mape)` for the CPI 12-steps ahead forecast and `mean(cpi_error_1$mape)` for the CPI 1-step ahead forecast, and `mean(ur_error_12$mape)` for the UR 12-steps ahead forecast and `mean(ur_error_1$mape)`for the UR 1-step ahead forecast.

Note: The CPI Series includes several observations where the CPI equals 0, so we had to adjust the series in order to calculate the MAPE. Therefore, the MAPE plot for CPI may be inaccurate.

Now test your model using a moving window backtesting scheme. Forecast out 12-steps ahead at each iteration, and plot the forecast errors observed at each iteration. Repeat for a 1-step ahead forecast horizon. Provide plots of both.

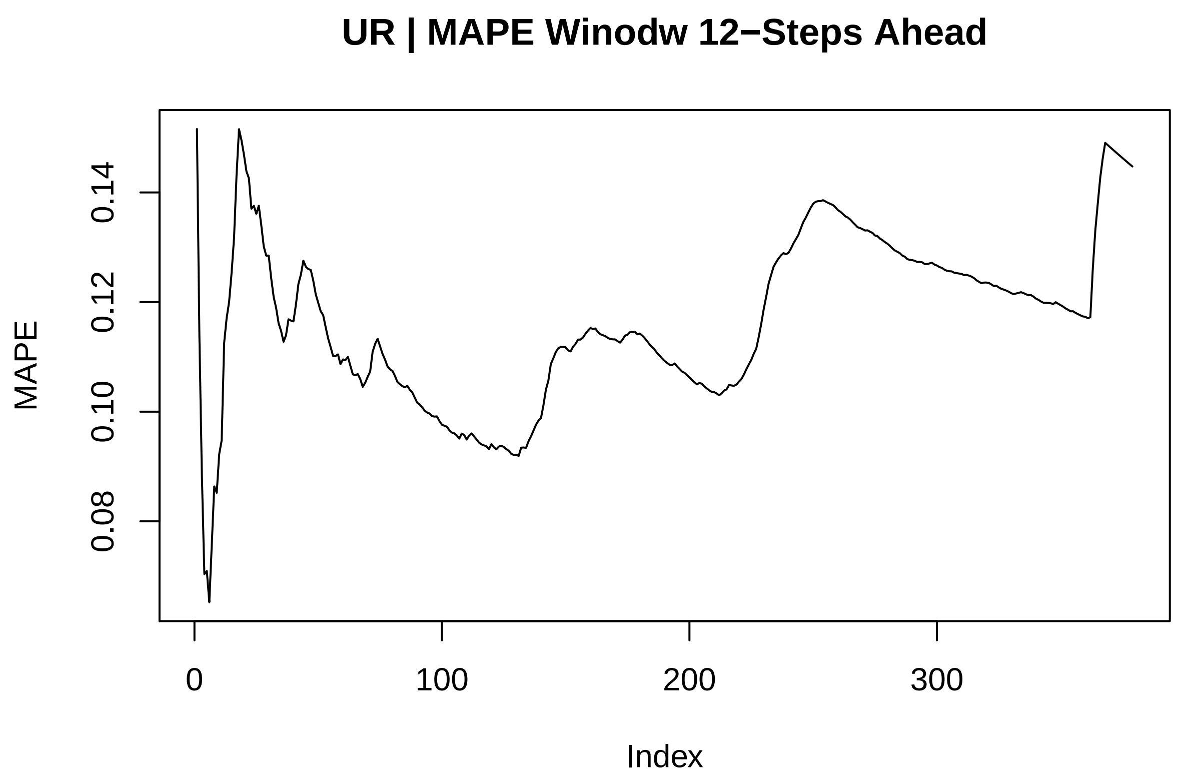
```r
window_backtesting <- function(model, data, orig, h, xreg=NULL,fixed = NULL, inc.mean = TRUE,
                               reest = 1){
  if(!inherits(data,"ts"))stop("data must be a time series object")
  arma_order <- model$arma
  regor = arma_order[c(1, 6, 2)]
  seaor = list(order = arma_order[c(3, 7, 4)],  period = arma_order[5])
  T = length(data)
  if (orig > T)
    orig = T
  if (h < 1)
    h = 1
  rmse = numeric(h)
  mabso = numeric(h)

  nori = T - orig
  err = matrix(0, nori, h)
  fcst = matrix(0, nori, h)
  jlast = T - 1
  time_vec <- time(data)
  ireest <- reest
  for (n in orig:jlast) {
    jcnt = n - orig + 1
    x <- window(data, time_vec[jcnt], time_vec[n])
    if (is.null(xreg))
      pretor = NULL
    else pretor = xre[jcnt:n]
    if (ireest == reest) {
      mm = arima(x, order = regor, seasonal = seaor, xreg = pretor,
                 fixed = fixed, include.mean = inc.mean)
      ireest <- 0
    }
    else {
      ireest <- ireest + 1
    }
    if (is.null(xreg)) {
      nx = NULL
    }
    else {
      nx = xreg[(n + 1):(n + h)]
    }
    fore = predict(mm, h, newxreg = nx)
    kk = min(T, (n + h))
    nof = kk - n
    pred = fore$pred[1:nof]
    obsd = data[(n + 1):kk]
    err[jcnt, 1:nof] = obsd - pred
    fcst[jcnt, 1:nof] = pred
  }
  for (i in 1:h) {
    iend = nori - i + 1
    tmp = err[1:iend, i]
    mabso[i] = sum(abs(tmp))/iend
    rmse[i] = sqrt(sum(tmp^2)/iend)
  }
  print("RMSE of out-of-sample forecasts")
  print(rmse)
  print("Mean absolute error of out-of-sample forecasts")
  print(mabso)
  backtest <- list(origin = orig, error = err, forecasts = fcst,
                   rmse = rmse, mabso = mabso, reest = reest)
}
```

```
# CPI 12-steps ahead
window_cpi_12 <- window_backtesting(cpi_final_model, CPIGRL_ts_mod, orig=350, h=12,
                                   xreg=NULL,fixed = NULL, inc.mean =TRUE, reest = 1)

cpi_error_12.2 <- data.frame(window_cpi_12$error)
cpi_error_12.2$actual <- CPIGRL_ts_mod[351:729]
cpi_error_12.2$abs <- abs(cpi_error_12.2$X12/cpi_error_12.2$actual)
cpi_error_12.2$mape <- cummean(cpi_error_12.2$abs)

plot(cpi_error_12.2$mape, type='l', main= "CPI | MAPE Winodw 12-Steps Ahead", ylab="MAPE")

# CPI 1-steps ahead
window_cpi_1 <- window_backtesting(cpi_final_model, CPIGRL_ts_mod, orig=350, h=1,
                                   xreg=NULL,fixed = NULL, inc.mean =TRUE, reest = 1)

cpi_error_1.2 <- data.frame(window_cpi_1$error)
cpi_error_1.2$actual <- CPIGRL_ts_mod[351:729]
cpi_error_1.2$abs <- abs(cpi_error_1.2$window_cpi_1.error/cpi_error_1.2$actual)
cpi_error_1.2$mape <- cummean(cpi_error_1.2$abs)

plot(cpi_error_1.2$mape, type='l', main= "CPI | MAPE Winodw 1-Step Ahead", ylab="MAPE")
```

```
# UR 12-steps ahead
window_ur_12 <- window_backtesting(ur_final_model, UNRATE_ts, orig=350, h=12,
                                   xreg=NULL,fixed = NULL, inc.mean =TRUE, reest = 1)
```

```
## [1] "RMSE of out-of-sample forecasts"
##  [1] 0.8621538 1.2412884 1.4358392 1.6139843 1.4101294 1.1835463 1.2006695
##  [8] 1.2570708 1.2756988 1.3339234 1.3625754 1.4361881
## [1] "Mean absolute error of out-of-sample forecasts"
##  [1] 0.3130111 0.4187845 0.4521559 0.5341572 0.5385048 0.5573597 0.5938612
##  [8] 0.6447183 0.6583815 0.7054573 0.7374386 0.8044363
```

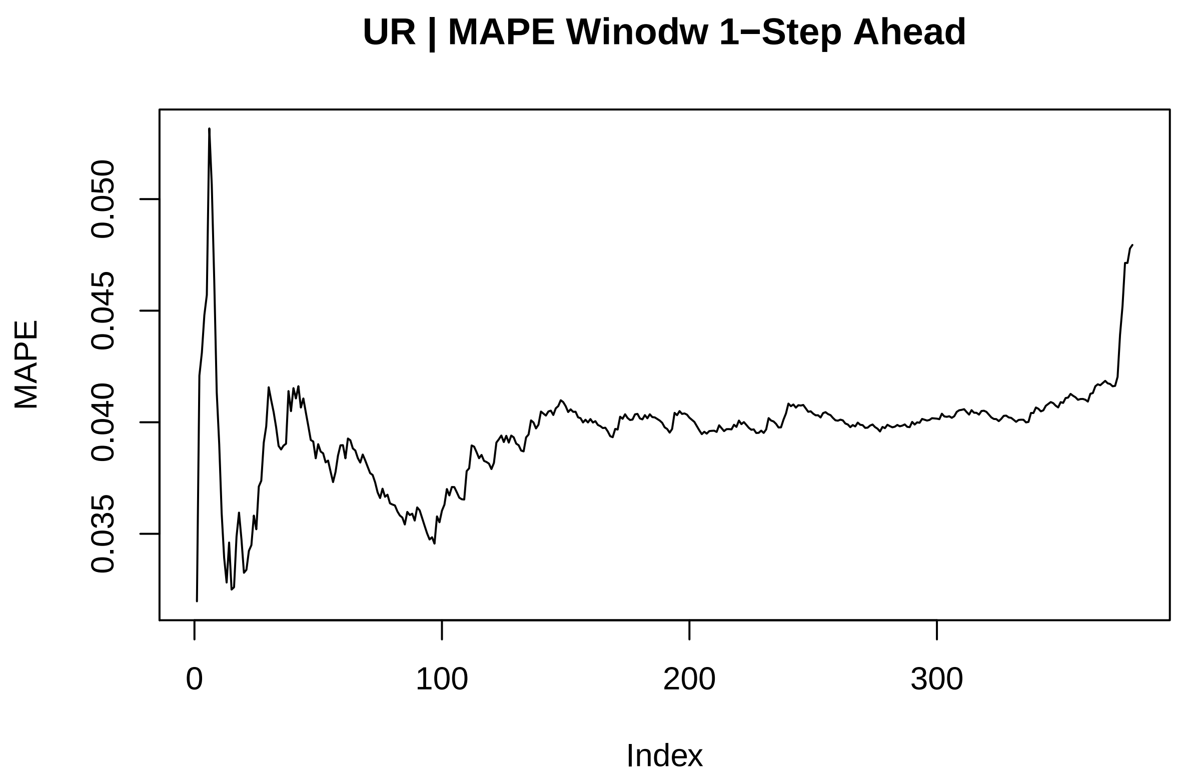
```
ur_error_12.2 <- data.frame(window_ur_12$error)
ur_error_12.2$actual <- UNRATE_ts[351:729]
ur_error_12.2$abs <- abs(ur_error_12.2$X12/ur_error_12.2$actual)
ur_error_12.2$mape <- cummean(ur_error_12.2$abs)

plot(ur_error_12.2$mape, type='l', main= "UR | MAPE Winodw 12-Steps Ahead", ylab="MAPE")
```



```
# UR 1-steps ahead
window_ur_1 <- window_backtesting(ur_final_model, UNRATE_ts, orig=350, h=1,
                                   xreg=NULL,fixed = NULL, inc.mean =TRUE, reest = 1)
```

```
## [1] "RMSE of out-of-sample forecasts"
## [1] 0.8621538
## [1] "Mean absolute error of out-of-sample forecasts"
## [1] 0.3130111
```

```r
ur_error_1.2 <- data.frame(window_ur_1$error)
ur_error_1.2$actual <- UNRATE_ts[351:729]
ur_error_1.2$abs <- abs(ur_error_1.2$window_ur_1.error/ur_error_1.2$actual)
ur_error_1.2$mape <- cummean(ur_error_1.2$abs)

plot(ur_error_1.2$mape, type='l', main= "UR | MAPE Winodw 1-Step Ahead", ylab="MAPE")
```



How do the errors found using a recursive backtesting scheme compare with the errors observed using a moving average backtesting scheme? Which scheme showed higher errors overall, and what does that tell you about your model?

Since the MAPE of the CPI is potentially inaccurate, we focus on the unemployment rate here.

```r
cat("Recrusieve Mean MAPE UR 12 Steps:", mean(ur_error_12$mape),"\n")
```

```
## Recrusieve Mean MAPE UR 12 Steps: 0.1153935
```

```r
cat("Recrusieve Mean MAPE UR 1 Step  :", mean(ur_error_1$mape),"\n")
```

```
## Recrusieve Mean MAPE UR 1 Step  : 0.03877583
```

```r
cat("Window Mean MAPE UR 12 Steps    :", mean(ur_error_12.2$mape),"\n")
```

```
## Window Mean MAPE UR 12 Steps    : 0.1165728
```

```r
cat("Window Mean MAPE UR 1 Step      :", mean(ur_error_1.2$mape),"\n")
```

```
## Window Mean MAPE UR 1 Step      : 0.03956734
```

Overall, it seems that the recursive backtesting scheme is more accurate. Our model performs worse if it has a shorter input timeframe.

III. Conclusion and Future Work

In conclusion it can be said, that the singularity of COVID-19 renders our forecasts for the unemployment rate relatively useless. The forecasts for the consumer price index seem more realistic. While we found Granger causality between CPI and UR it was not in the direction that we would have expected. For future work, it would be interesting to fit other models (e.g. Exponential Smoothing algorithm) to these data and evaluate their performance.

Interestingly, our VAR model seems to predict that based on our current data, the CPI growth rate seems to Granger cause the unemployment rates, though the unemployment rates seem to have no as strong of a reverse causal effect on CPI growth rates. This could be extended to analyze the Phillips curve model and it's implications in both the short run and the long run.

IV. References

Data

Organization for Economic Co-operation and Development, Consumer Price Index: Total All Items for the United States [CPALTT01USM657N], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/CPALTT01USM657N, December 8, 2020.

United States. Bureau of Labor Statistics, Unemployment Rate [UNRATENSA], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/UNRATENSA, December 8, 2020.

Other References

Blinder, Alan S. 2018. "Is the Phillips Curve Dead? And Other Questions for the Fed." *The Wall Street Journal*, May 3, 2018. Available at: https://www.wsj.com/articles/is-the-phillips-curve-dead-and-other-questions-for-the-fed-1525388237.

Federal Reserve Board of Governors. 2018. "Monetary Policy Principles and Practice." March 08, 2018. Available at: https://www.federalreserve.gov/monetarypolicy/monetary-policy-what-are-its-goals-how-does-it-work.htm#:~:text=The%20Federal%20Reserve%20Act%20mandates,for%20monetary%

Ng, Michael, David Wessel, and Louise Sheiner. 2018. "The Hutchins Center Explains: The Phillips Curve." *The Brookings Institution*, August 21, 2018. Avaialbe at: https://www.brookings.edu/blog/up-front/2018/08/21/the-hutchins-center-explains-the-phillips-curve/.

Steelman, Aaron. 2011. "The Federal Reserve's 'Dual Mandate': The Evolution of an Idea." *Federal Reserve Bank of Richmond*, Economic Brief, December 2011, No. 11-12. Available at: https://www.richmondfed.org/publications/research/economic_brief/2011/eb_11-12.

United States. Congress. 1977. *Federal Reserve Reform Act of 1977.* Public Law 95-188—Nov. 16, 1977. Available at: https://fraser.stlouisfed.org/title/federal-reserve-reform-act-1977-1040.