Lab3

Jayashree Raman / Ted Pham / Phat Doan

Question 1:

During your EDA, you notice that your data exhibits both seasonality (different months have different heights) AND that there is a clear linear trend. How many order of non-seasonal and seasonal differencing would it take to make this time-series stationary in the mean? Why?

Unemployment Rate Data

Import libraries

```
suppressWarnings(library(Hmisc))
```

```
## Loading required package: lattice
## Loading required package: survival
## Loading required package: Formula
## Loading required package: ggplot2
##
## Attaching package: 'Hmisc'
## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
```

```
suppressWarnings(library(forecast))
suppressWarnings(library(tseries))
suppressWarnings(library(astsa))
```

```
##
## Attaching package: 'astsa'
## The following object is masked from 'package:forecast':
##
##     gas
```

```
suppressWarnings(library(vars))
```

```
## Loading required package: MASS
## Loading required package: strucchange
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich

## Loading required package: urca

## Loading required package: lmtest
```

```
suppressWarnings(library(psych))
```

```
##
## Attaching package: 'psych'

## The following object is masked from 'package:Hmisc':
##
##     describe

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
suppressWarnings(library(zoo))
```

Exploratory Data Analysis

```
df <- read.csv("UNRATENSA.csv", stringsAsFactors = FALSE)

summary(df)
```

```
##      DATE             UNRATENSA
##  Length:834         Min.   : 2.400
##  Class :character   1st Qu.: 4.700
##  Mode  :character   Median : 5.600
##                     Mean   : 5.801
##                     3rd Qu.: 6.900
##                     Max.   :11.400
```

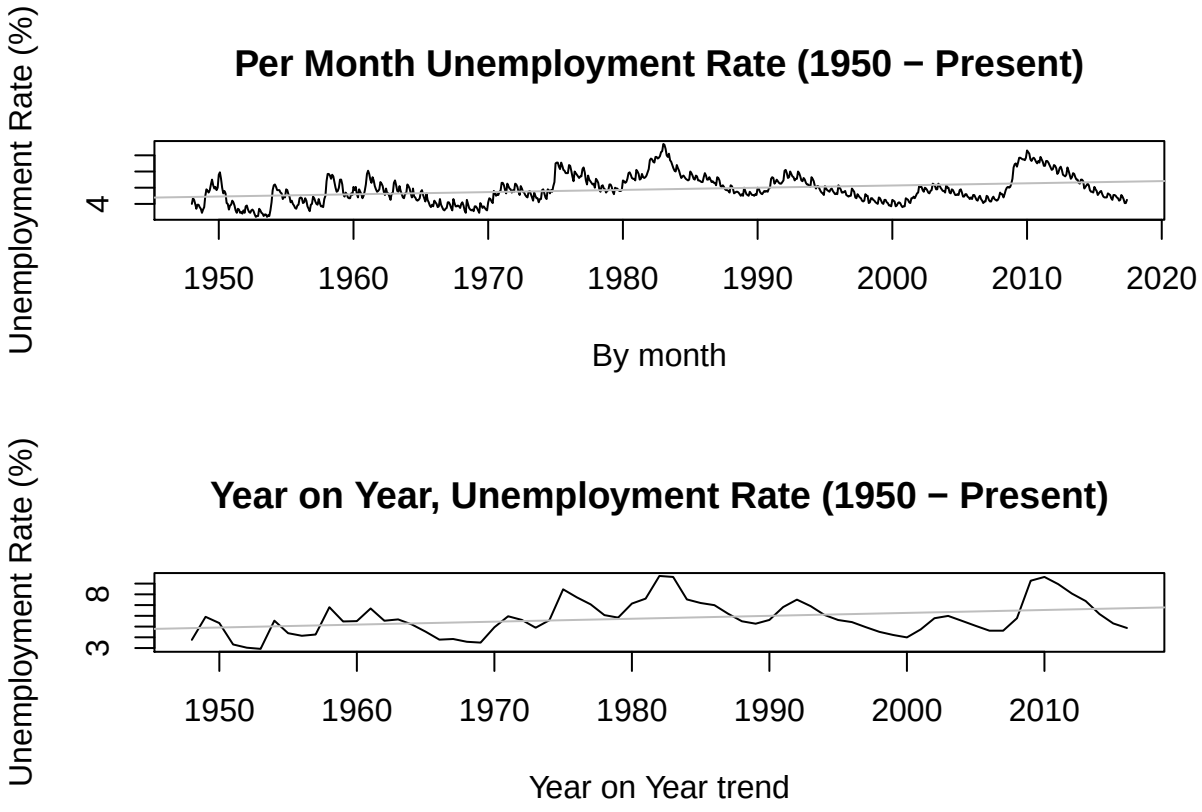
```
describe(df)
```

```
## Warning: NAs introduced by coercion

## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning
## Inf

## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning
## -Inf
```

```
##           vars   n mean   sd median trimmed  mad min  max range skew
## DATE*        1 834  NaN   NA     NA     NaN   NA Inf -Inf  -Inf   NA
## UNRATENSA    2 834  5.8 1.68    5.6    5.69 1.63 2.4 11.4     9 0.58
##           kurtosis   se
## DATE*           NA   NA
## UNRATENSA     0.06 0.06
```

```
unratensa <- ts(df$UNRATENSA, frequency = 12, start = c(1948,1))

par(mfrow=c(2,1))
plot(unratensa, main = "Per Month Unemployment Rate (1950 - Present)",
      xlab = "By month", ylab = "Unemployment Rate (%)")
abline(reg=lm(unratensa~time(unratensa)), col="grey")
```

```r
#Display Year on Year trend
plot(aggregate(unratensa,FUN=mean),  main = "Year on Year, Unemployment Rate (1950 - Present)",
      xlab = "Year on Year trend", ylab = "Unemployment Rate (%)")
abline(reg=lm(unratensa~time(unratensa)), col="grey")
```




```r
par(mfrow=c(1,1))
#
boxplot(unratensa~cycle(unratensa), main = "Unemployment Rate by Month (1950 - Present)",
      xlab = "Month", ylab = "Unemployment Rate (%)")
```

The unemployment rate data spans from January 1948 to June 2017, accounting for 834 months. There was no missing value in the data. The lowest and higest rate are within percentage range [2.4,11.4] corresponding to 1952-10-01 and 1983-01-01.

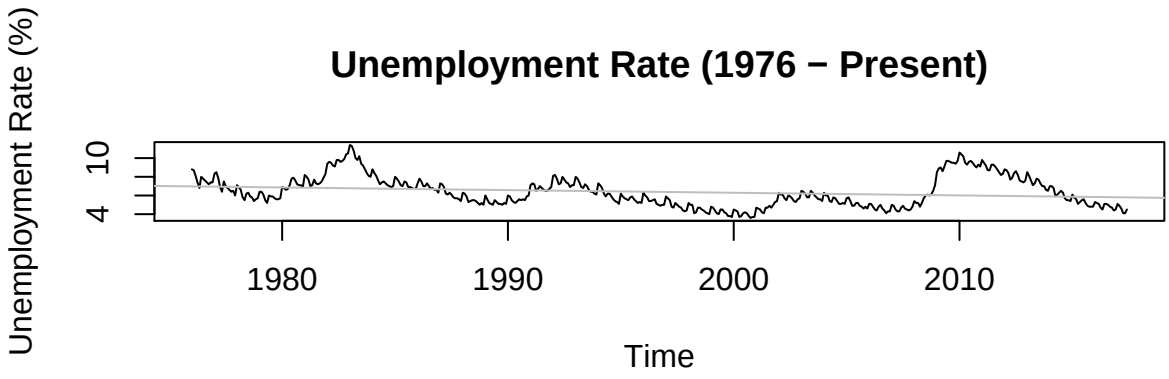
Important Inferences: 1/ The year on year trend shows slight increasing trend of unemployment rates over the year. 2/ The variance and the mean value in Febuary and June is much higher than rest of the months. 3/ Even though the mean value of each month is quite different their variance is small. Hence, we have strong seasonal effect with a cycle of 12 months or less.

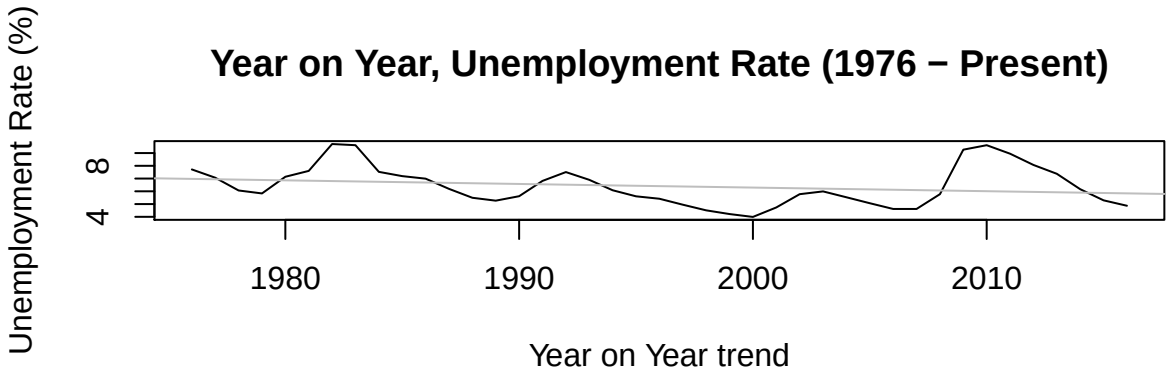
Shorten data timeframe

Between 1948 and 1970, the unemployment rate has a lower variance compared to 1970 to present. We observe lowest rate 2.1 was in October 1953. This rate seemed low and can be argued unreasonable for this current time period. There were several technological advances made in the 1970s that significantly changed the way society functioned. The first personal computer was introduced, internet was developed, rapid job automation through robotic breakthrough, the wolrd becomes more interconnected. These disruptions makes labor market more volatile, especially for the U.S. market with automation and offshoring that rapidly reducing the workforce without new job replacement. However, others can also argue that while blue labor jobs are disappearing, other jobs are also created. Therefore as a group, we have decided to forcus on 1970s (i.e. from 1976) data forward to take into consideration the tectonic paradigm shift within the labor market. In addition, this will make the unemployment data consistent with the data for autosale that starts from 1976.

```
unrate.short <- window(unratensa,start = c(1976,1))

par(mfrow=c(2,1))
```

```r
plot(unrate.short, main = "Unemployment Rate (1976 - Present)",
      xlab = "Time", ylab = "Unemployment Rate (%)")
abline(reg=lm(unrate.short~time(unrate.short)), col="grey")

#Display Year on Year trend
plot(aggregate(unrate.short,FUN=mean),  main = "Year on Year, Unemployment Rate (1976 - Present)",
      xlab = "Year on Year trend", ylab = "Unemployment Rate (%)")
abline(reg=lm(unrate.short~time(unrate.short)), col="grey")
```



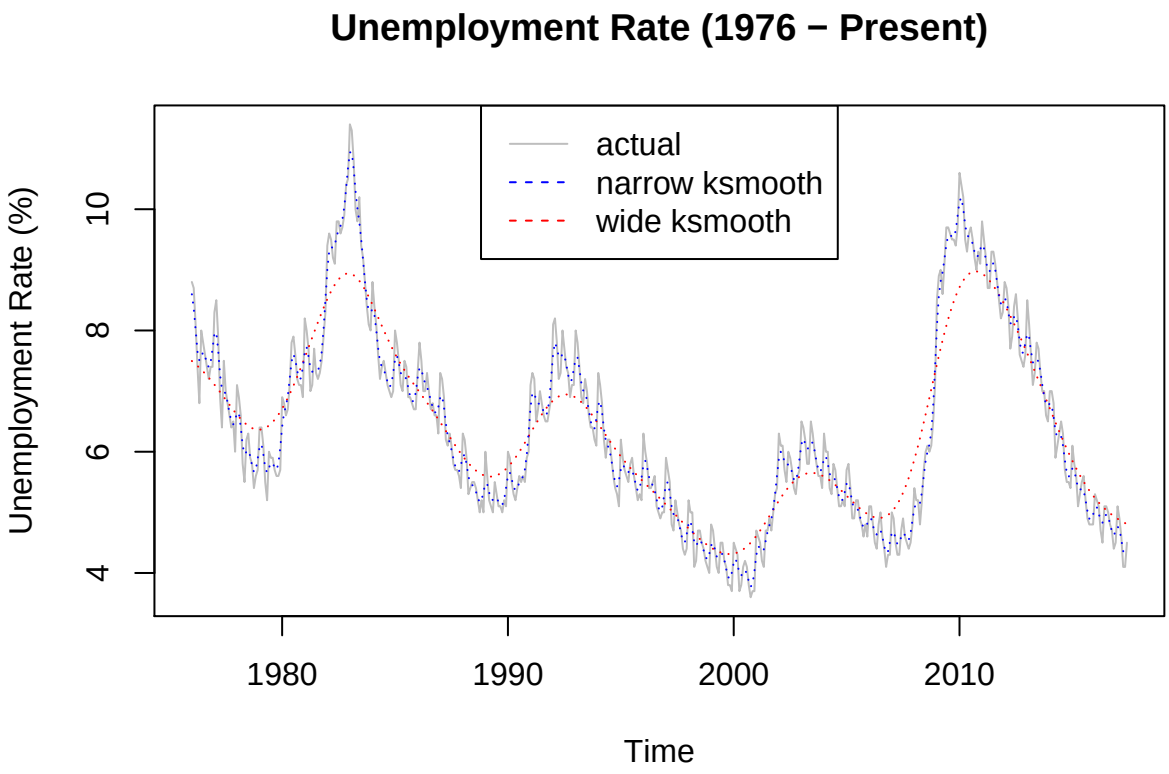



```r
par(mfrow=c(1,1))
#
boxplot(unrate.short~cycle(unrate.short), main = "Unemployment Rate by Month (1976 - Present)",
      xlab = "Month", ylab = "Unemployment Rate (%)")
```

Important Inferences: 1/ The year on year trend shows slight decreasing trend of unemployment rates over the year. 2/ The variance and the mean value for November is much less than rest of the months.

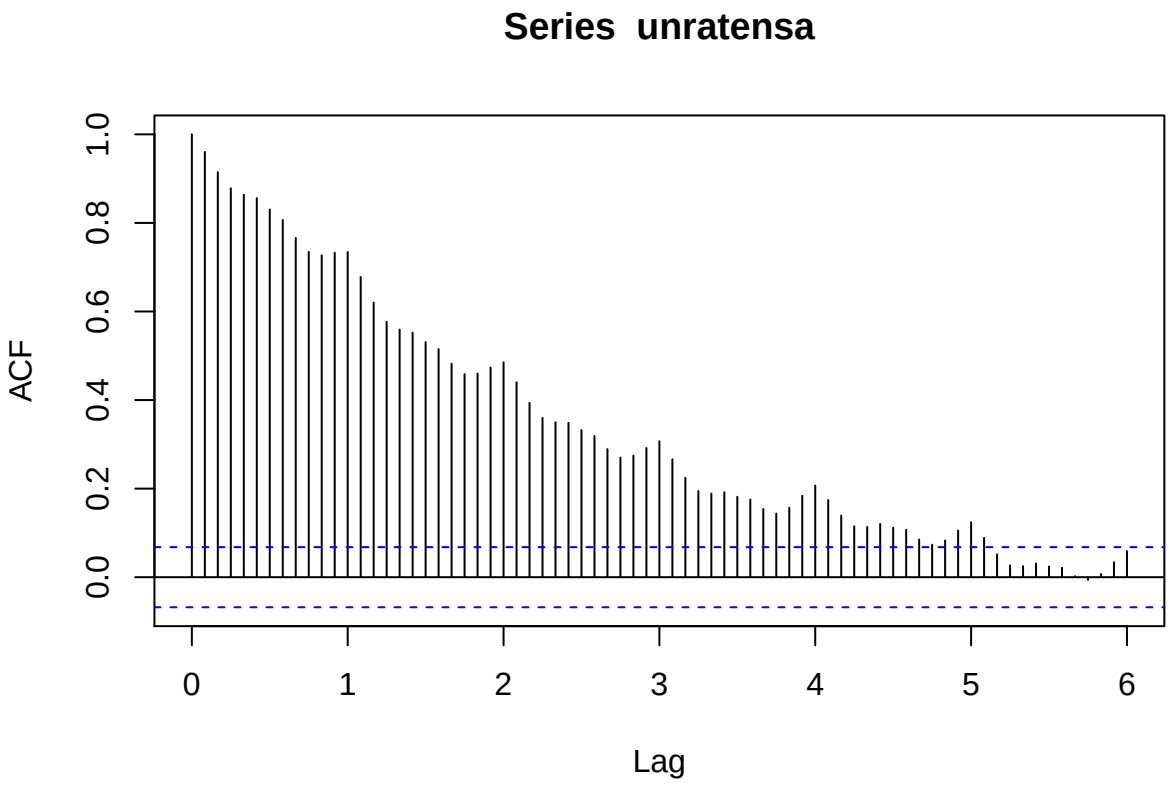
Trend and Seasonality

```
# plotting trend and seasonality
k.smooth.wide <- ksmooth(time(unrate.short), unrate.short, kernel = c("normal"), bandwidth = 3)
k.smooth.narrow <- ksmooth(time(unrate.short), unrate.short, kernel = c("normal"), bandwidth = 0.3)
plot(unrate.short, col = 'gray', main = "Unemployment Rate (1976 - Present)",
      xlab = "Time", ylab = "Unemployment Rate (%)")
lines(k.smooth.wide$x, k.smooth.wide$y, col =  'red',lty = 3) #trend
lines(k.smooth.narrow$x, k.smooth.narrow$y, col =  'blue', lty = 3) #seasonality
legend("top", lty = c(1,2,2), legend = c("actual","narrow ksmooth","wide ksmooth"), col = c("gray","blu
```

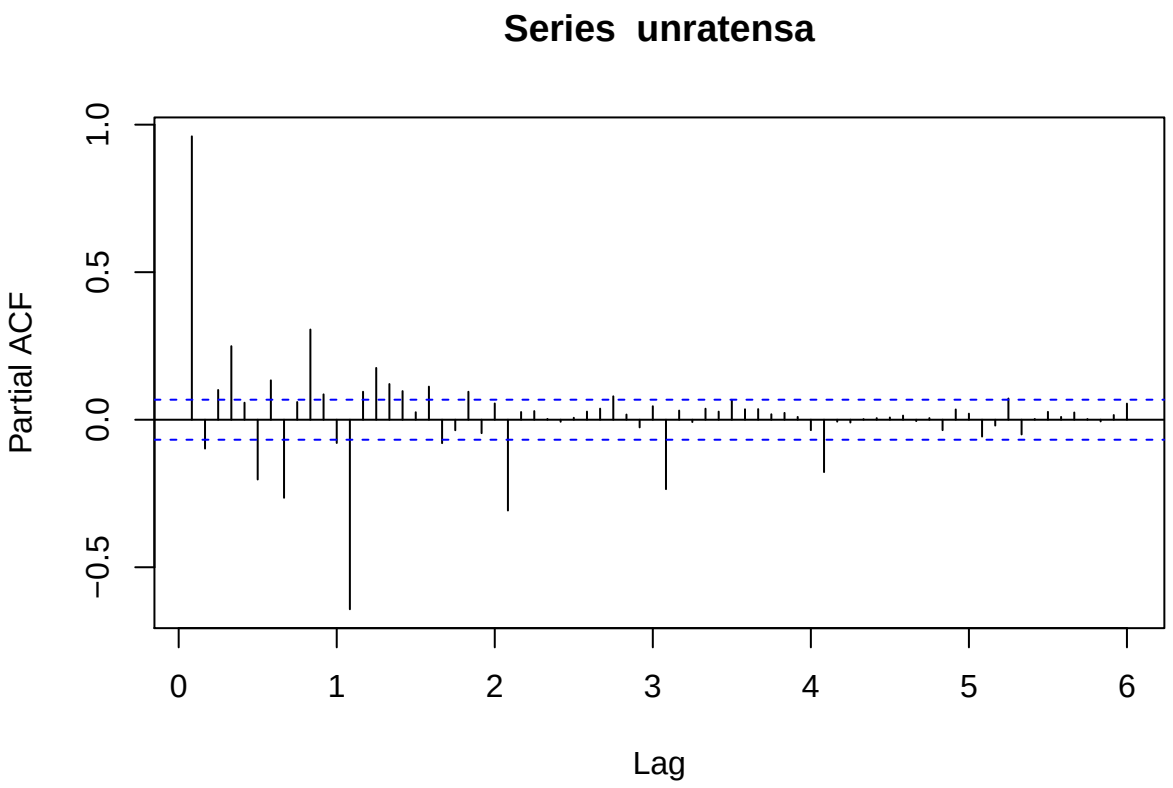


Using a narrow kernel smoother, we see evidence of seasonality in the blue line and the underlying trend in the red line. We can further confirm the appearance of seasonality and trend with acf and pacf plots.

```
#par(mfrow = c(2,1))
acf(unratensa, lag.max = 72)
```




```
pacf(unratensa, lag.max = 72)
```

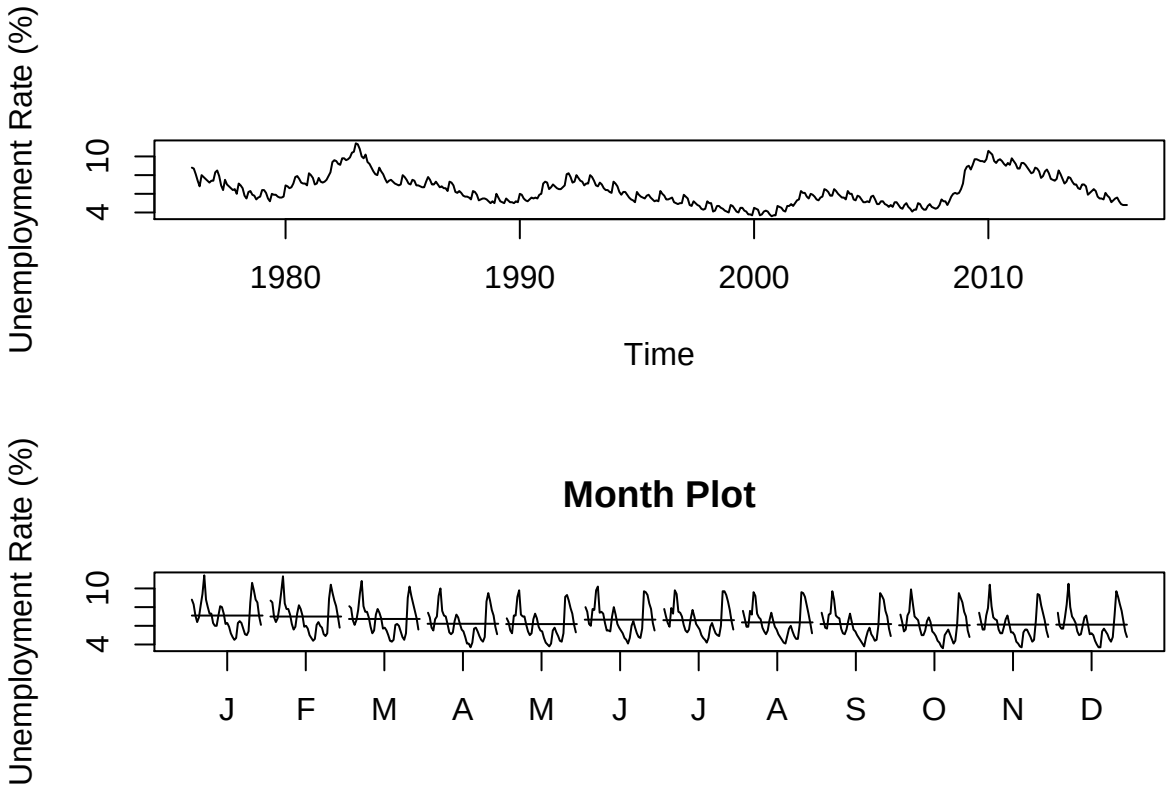


From the acf, pacf plots, we see clear evidence of AR process (with pacf peak at 1 month lag 0.1 and gradual decrease of acf) and seasonality with pcaf peaked again at lag 1 = 12 month. From the ACF plot, a gradual decrease of ACF over time also indicates trends. Additionaly, we observe sporadic uptick at lags 1, 2, 3, which is an evident of seasonality every 12 months.

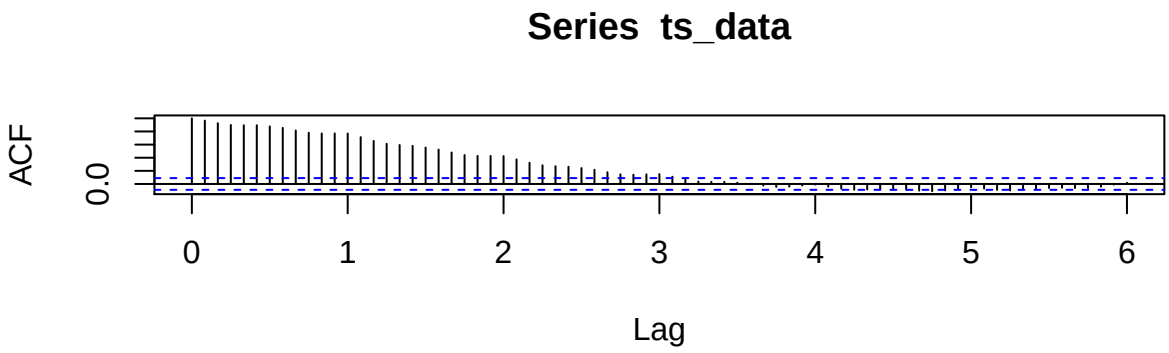
Conclusion: The trend and seasonality must be omitted from the time series

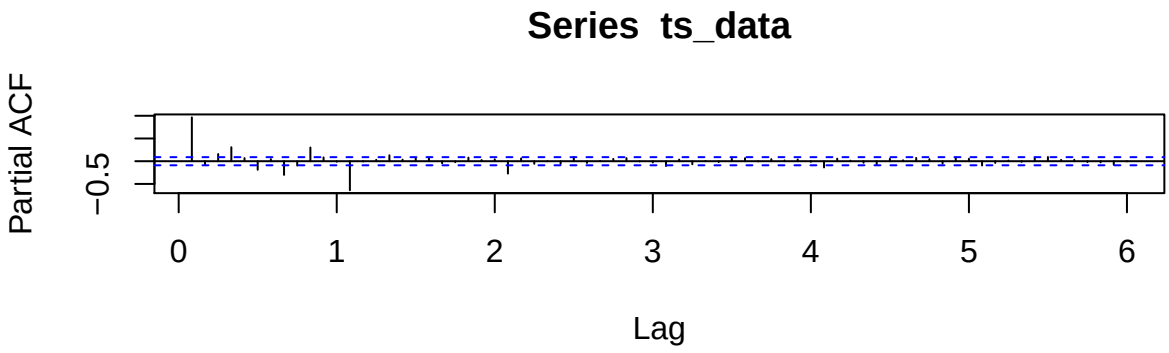
Transformation to stationary time series

```
print_tsplots <- function(ts_data) {
  par(mfrow=c(2,1))
  plot(ts_data, ylab="Unemployment Rate (%)")
  monthplot(ts_data, ylab="Unemployment Rate (%)", main="Month Plot")

  par(mfrow=c(2,1))
  acf(ts_data,lag.max=72)
  pacf(ts_data,lag.max=72)
}

emp.training <- window(unratensa, start = c(1976,1), end = c(2015,12))
emp.test <- window(unratensa, start = c(2016,1))

print_tsplots(emp.training)
```


Unemployment Rate (%)
4
10
1980
1990
2000
2010
Time

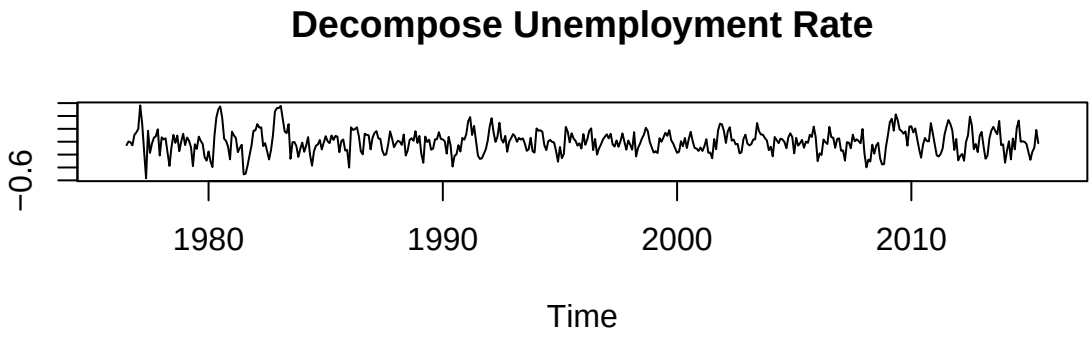
Month Plot
Unemployment Rate (%)
4
10
J
F
M
A
M
J
J
A
S
O
N
D

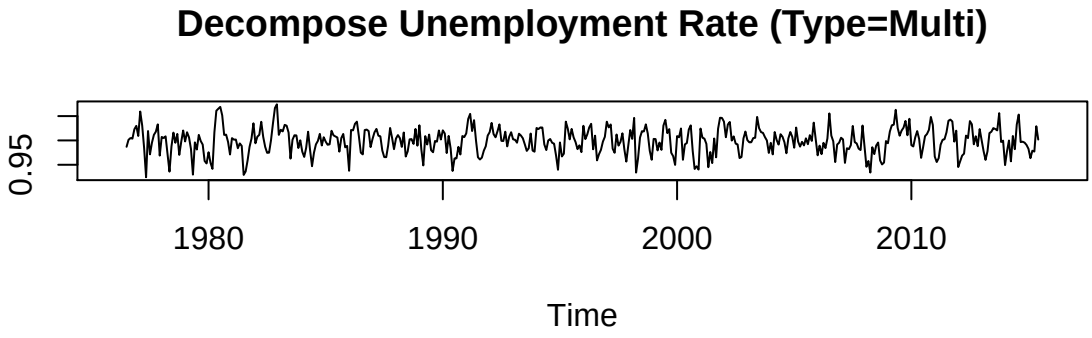




From the time plot, we observe random walks in the data so it'd be reasonable to use first difference to stationary the data. Also from the adf plot, the data exhibits seasonality at 12 months (lag=1) so D can also be 1. Before we perform the difference, we need to determine if a non-linear transformation is necessary. To do this, we examine the relationship between the trend and seasonality, specifically whether it's additive or multiplicative.

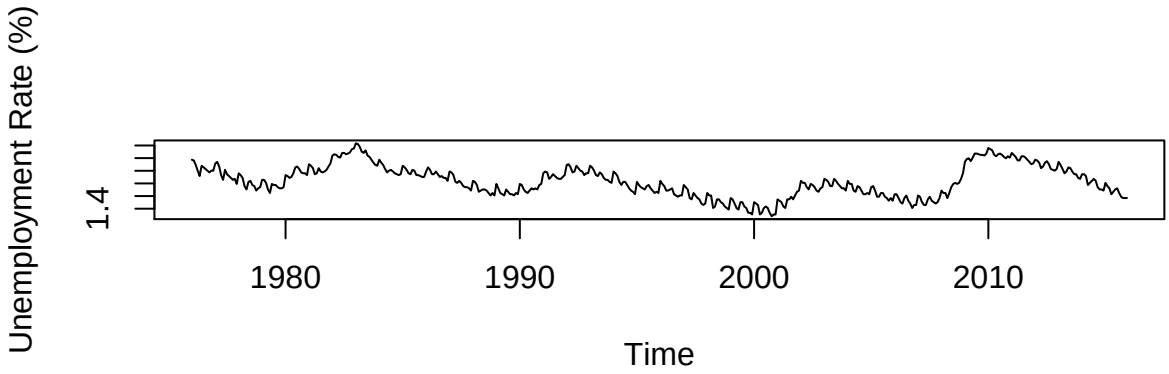
```
par(mfrow=c(2,1))
plot(decompose(emp.training)$random, ylab="", main="Decompose Unemployment Rate")
plot(decompose(emp.training,type='multi')$random, ylab="", main="Decompose Unemployment Rate (Type=Multi
```

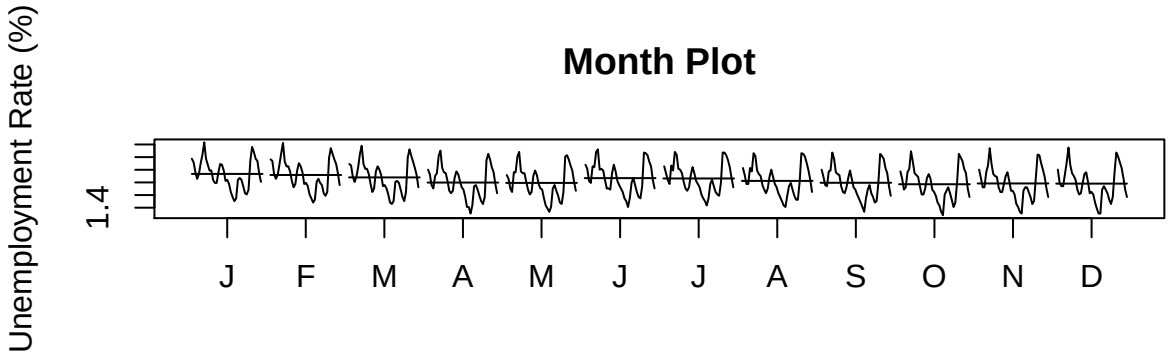


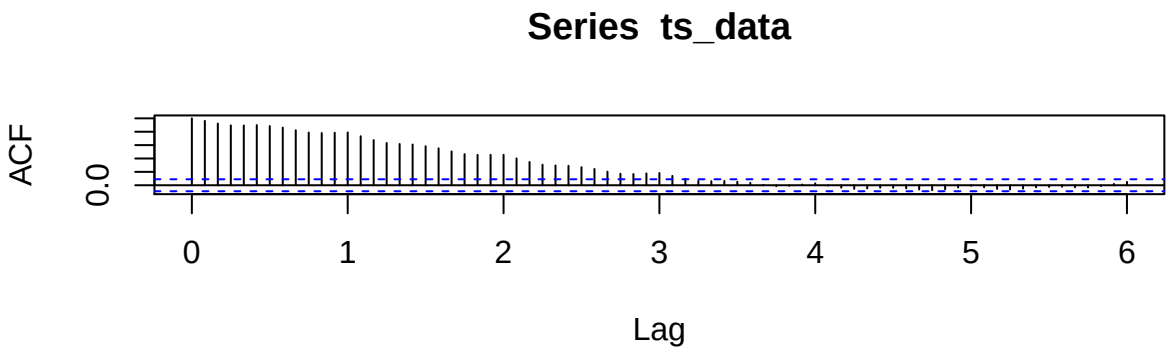


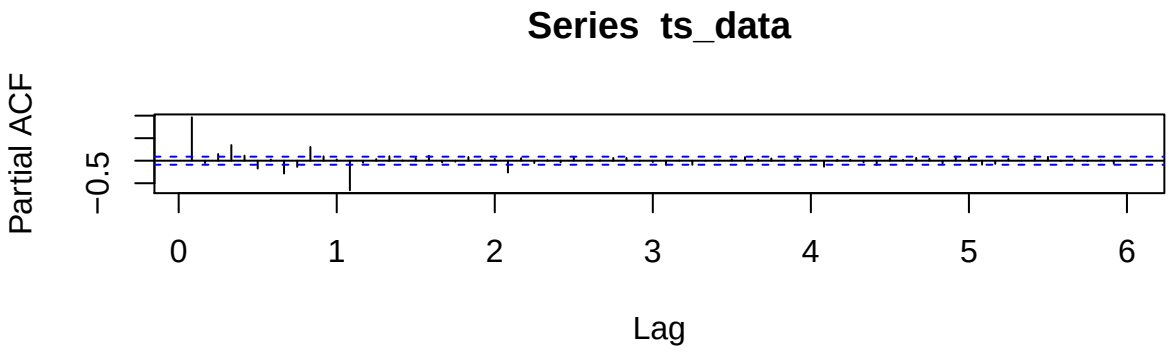
From the decomposition plots of the random component, it seems the multiplicative model for trend and seasonality is better because of a more constant variance. A log transfomration would be useful here.

```
emp.log <- log(emp.training)
print_tsplots(emp.log)
```


Unemployment Rate (%)
1.4
1980
1990
2000
2010
Time


Month Plot
Unemployment Rate (%)
1.4
J
F
M
A
M
J
J
A
S
O
N
D

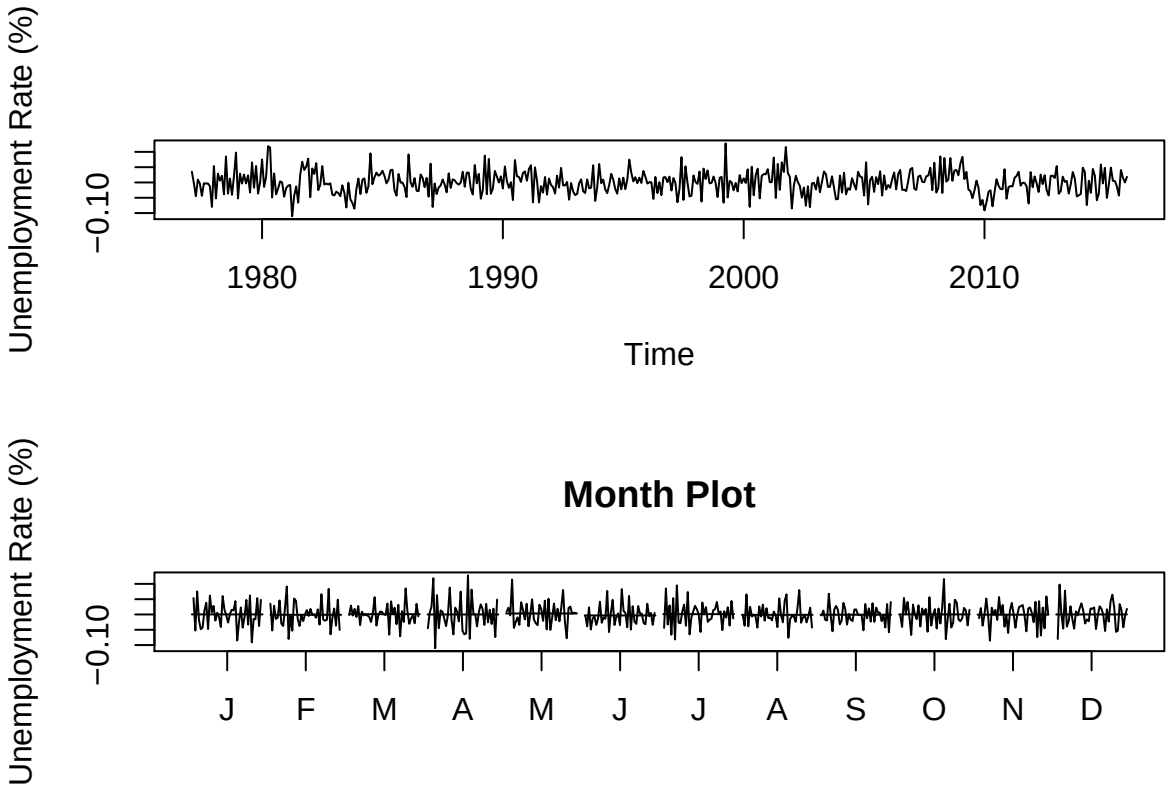


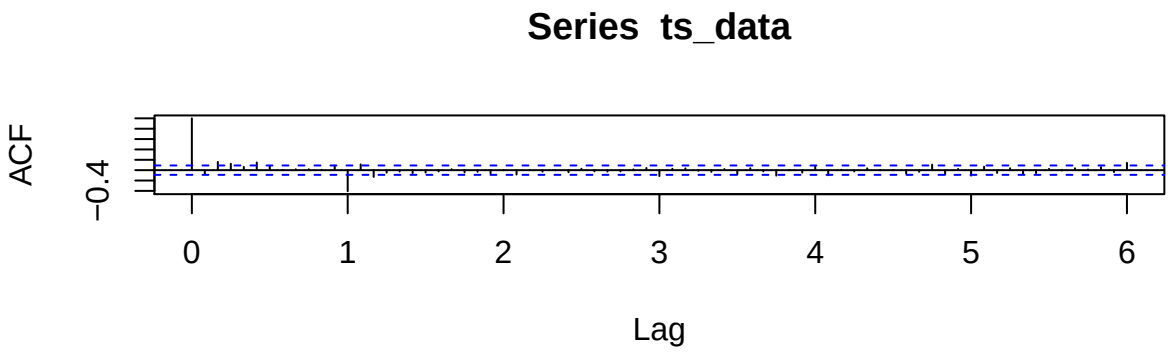


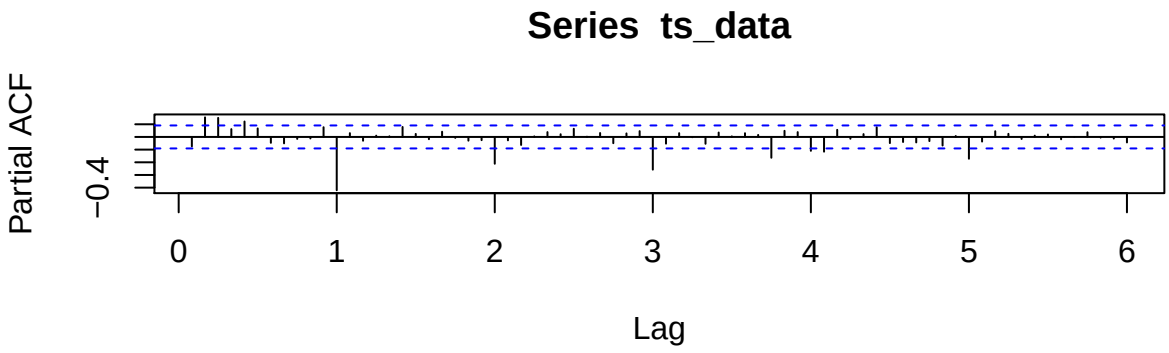

```
emp.test.log <- log(emp.test)
```

Now we take the difference to substract the random walk effect and deseason the data.

```
emp.yd.log <- diff(diff(emp.log, lag = 12))
print_tsplots(emp.yd.log)
```


Unemployment Rate (%)
−0.10
1980
1990
2000
2010
Time
Unemployment Rate (%)
Month Plot
−0.10
J
F
M
A
M
J
J
A
S
O
N
D





The pacf still have values outside of the confidence interval boundaries. We then run the adf and the unit root tests.

```
adf.test(emp.log)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  emp.log
## Dickey-Fuller = -2.4449, Lag order = 7, p-value = 0.3899
## alternative hypothesis: stationary
```

```
pp.test(emp.log)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  emp.log
## Dickey-Fuller Z(alpha) = -10.733, Truncation lag parameter = 5,
## p-value = 0.5107
## alternative hypothesis: stationary
```

```
adf.test(emp.yd.log)
```

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

```
##
##  Augmented Dickey-Fuller Test
##
```

```
## data:  emp.yd.log
## Dickey-Fuller = -6.1878, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
pp.test(emp.yd.log)
```

```
## Warning in pp.test(emp.yd.log): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  emp.yd.log
## Dickey-Fuller Z(alpha) = -615.36, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

We see from the above tests that we need one differencing for the seasonal lag = 12(D=1) and one difference for the non-seasonal lags(d=1) to make the series weakly stationary - we will use these parameters later to create our SARIMA model.

The unit root test and the adf.test both provide evidence (p=0.01) to reject the null hypothesis that the transformed series is not stationary.

Question 2: SARIMA

It is Dec 31, 2016 and you work for a non-partisan think tank focusing on the state of the US economy. You are interested in forecasting the unemployment rate through 2017 (and then 2020) to use it as a benchmark against the incoming administration's economic performance. Use the dataset UNRATENSA.csv and answer the following:

A) Build a SARIMA model using the unemployment data and produce a 1 year forecast and then a 4 year forecast. Because it is Dec 31, 2016, leave out 2016 as your test data.

Modeling

We create a function to find the best p,q,P,Q based on lowest AIC score.

```
# define get best arima function
get.best.arima <- function(ts,p,q,P,Q)
{
  #initialize best.aic to a large number
  best.aic <- 1e8
   for (p.i in 0:p ) for (q.i in 0:q)
    for (P.i in 0:P)  for (Q.i in 0:Q)
    {
      try(fit <-Arima(ts, order = c(p.i,1, q.i), seasonal = list(order = c(P.i,1, Q.i)), method = "ML"))

      fit.aic <- fit$aic
      #fit.aic <- -2*fit$loglik + (log(n)+1)*length(fit$coef)
      if (fit.aic < best.aic)
      {
        best.aic <- fit.aic
        best.fit <-fit
```

```
        best.model <- c(p.i,1,q.i,P.i,1,Q.i)
        print(c(p.i,1,q.i,P.i,1,Q.i,best.aic, fit$bic, fit$rmse))
      }
    }
  list(best.aic,best.fit,best.model)
}
```

From the previous PACF plot, we set max values for p,q as 5,5.

```
get.best.arima(emp.log,5,5,1,1)
```

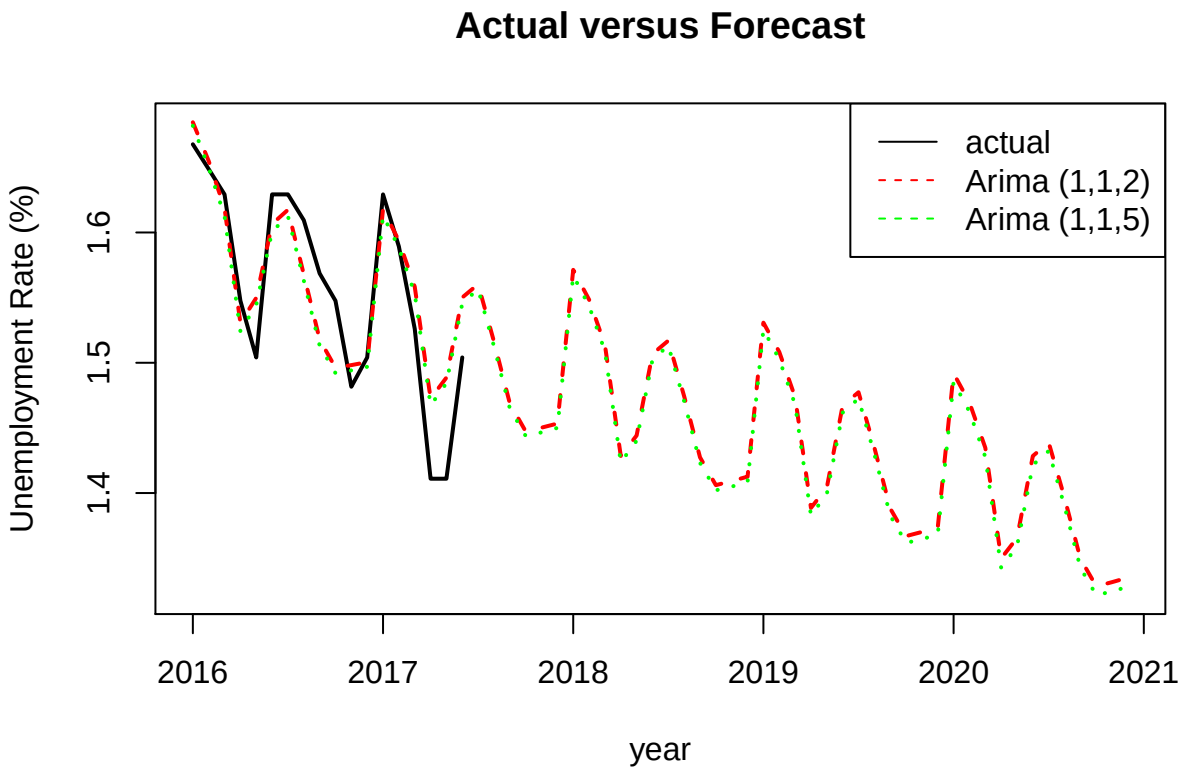
```
## [1]     0.000     1.000     0.000     0.000     1.000     0.000 -1724.307
## [8] -1720.160
## [1]     0.000     1.000     0.000     0.000     1.000     1.000 -1872.322
## [8] -1864.029
## [1]     0.000     1.000     0.000     1.000     1.000     1.000 -1885.012
## [8] -1872.573
## [1]     0.000     1.000     2.000     1.000     1.000     1.000 -1894.284
## [8] -1873.553
## [1]     0.000     1.000     4.000     1.000     1.000     1.000 -1894.626
## [8] -1865.602
## [1]     0.000     1.000     5.000     0.000     1.000     1.000 -1897.696
## [8] -1868.672
## [1]     0.000     1.000     5.000     1.000     1.000     1.000 -1907.982
## [8] -1874.812
## [1]     1.000     1.000     2.000     1.000     1.000     1.000 -1913.148
## [8] -1888.270
## [1]     1.000     1.000     5.000     1.000     1.000     1.000 -1914.739
## [8] -1877.422
```

```
## [[1]]
## [1] -1914.739
##
## [[2]]
## Series: ts
## ARIMA(1,1,5)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1     ma2      ma3     ma4     ma5    sar1     sma1
##       0.7472  -0.8466  0.1961  -0.0219  0.0083  0.1201  0.2063  -0.8730
## s.e.  0.0894   0.0978  0.0624   0.0638  0.0615  0.0562  0.0675   0.0489
##
## sigma^2 estimated as 0.0009229:  log likelihood=966.37
## AIC=-1914.74   AICc=-1914.34   BIC=-1877.42
##
## [[3]]
## [1] 1 1 5 1 1 1
```

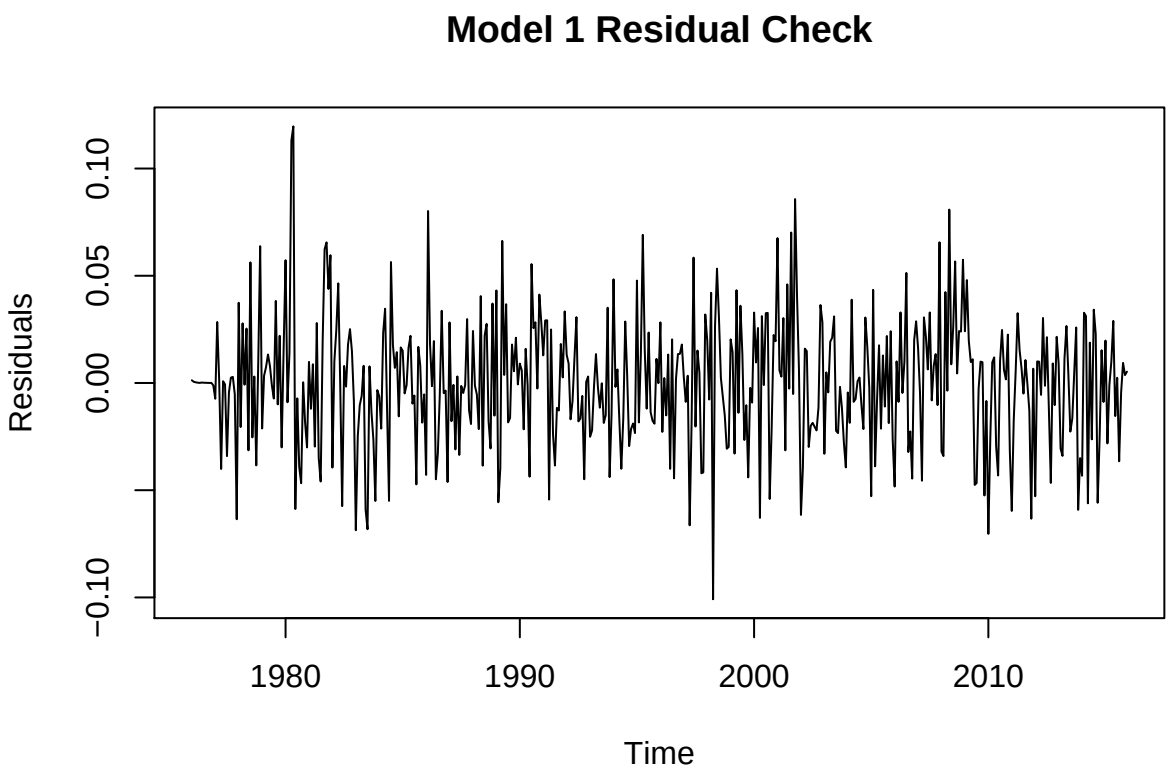
The output of our best.arima.model is (1,1,5,1,1,1) which has an AIC of -1914.7. The next best model is (1,1,2,1,1,1) with AIC of -1913.148. Since the AIC's are very close we now decide which model to move forward with.

```
m1 <- Arima(emp.log, order = c(1, 1, 2), seasonal = list(order = c(1, 1, 1)))
m2 <- Arima(emp.log, order = c(1, 1, 5), seasonal = list(order = c(1, 1, 1)))

forecast1 <- forecast(m1, h = length(emp.test.log)+42)
```

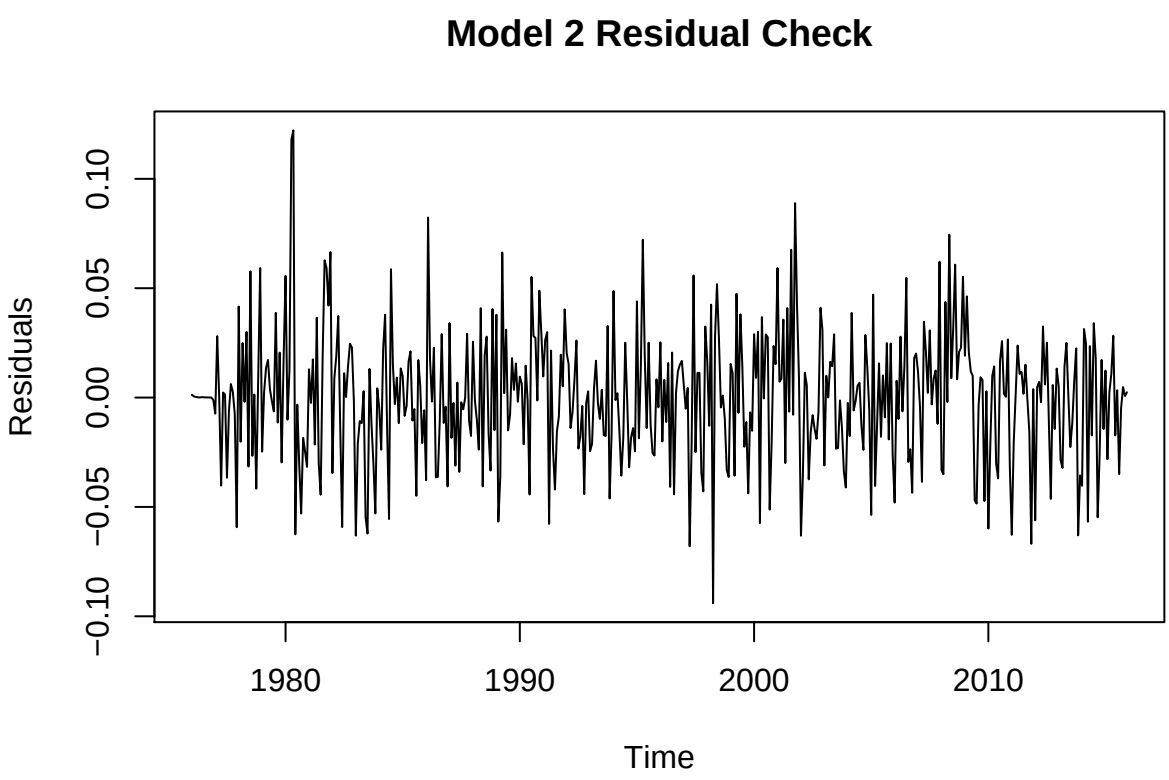
```r
forecast2 <- forecast(m2, h = length(emp.test.log)+42)

ts.plot(emp.test.log,forecast1$mean, forecast2$mean,
        gpars=list(main="Actual versus Forecast",xlab="year", ylab="Unemployment Rate (%)"),
        col = c("black", "red", "green"), lwd = 2, lty = 1:5)
legend("topright", lty = c(1,2,2), legend = c("actual","Arima (1,1,2)","Arima (1,1,5)"), col = c("black
```




```r
plot(m1$residuals,main="Model 1 Residual Check",ylab="Residuals")
```




```
plot(m2$residuals,main="Model 2 Residual Check",ylab="Residuals")
```




```
accuracy(m1)
```

```
##                        ME       RMSE        MAE        MPE     MAPE
## Training set 4.957169e-05 0.02995266 0.02305173 0.01250213 1.287594
##                    MASE        ACF1
## Training set 0.1940187 0.005626306
```

```
accuracy(m2)
```

```
##                        ME      RMSE       MAE        MPE     MAPE     MASE
## Training set 5.711522e-05 0.0297077 0.0228354 0.01129936 1.275576 0.192198
##                      ACF1
## Training set -0.001764256
```

Since the two models are extremely close, we decide to go with (p,d,q,P,D,Q) of (1,1,2,1,1,1) because of its relative simplicity compared to the other model.

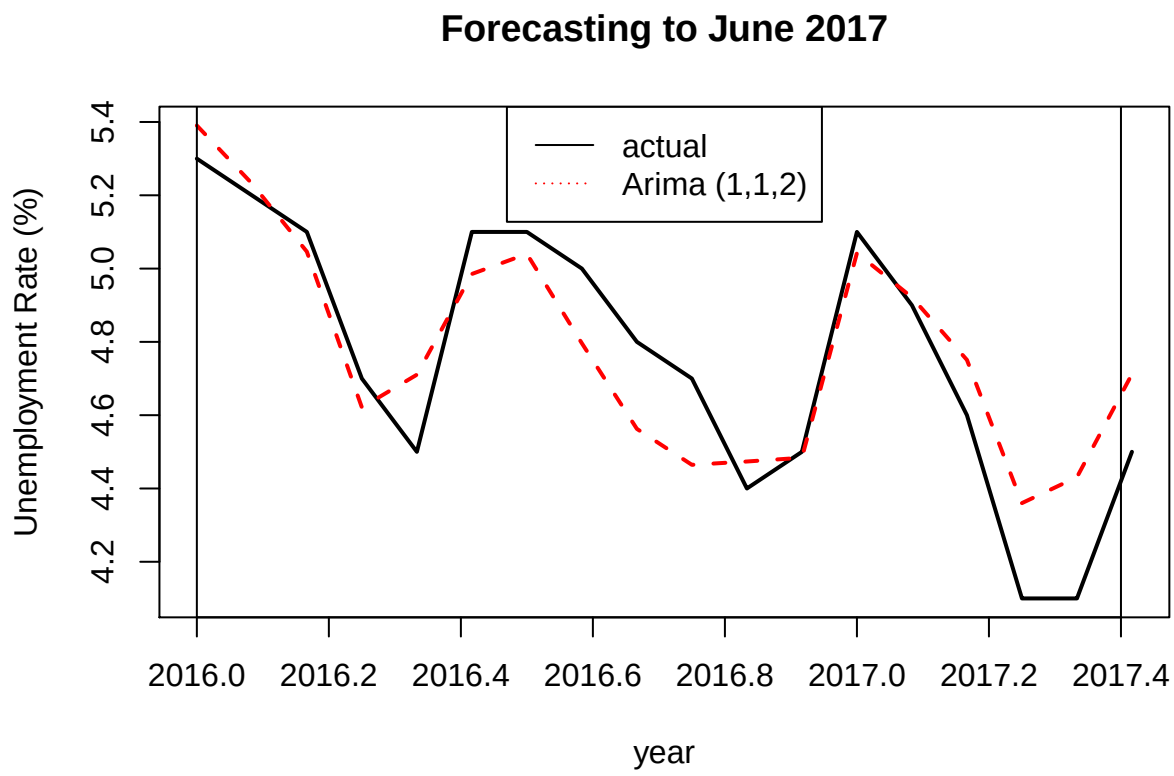
But first, some helper functions for plotting and calculating the root mean square error.

```
#helper function
print_resid_chart <- function(m) {

par(mfrow=c(2,2))
plot(m$residuals,ylab="Residuals")
hist(m$residuals,ylab="Residuals")
acf(m$residuals, 48)
pacf(m$residuals, 48)
}
```

```
rmse <- function(error)
{
    sqrt(mean(error^2))
}
```

Forecasting to June 2017

```
m1 <- Arima(emp.log, order = c(1, 1, 2), seasonal = list(order = c(1, 1,1)))

forecast1 <- forecast(m1, h = 18)

ts.plot(emp.test,exp(forecast1$mean),
        gpars=list(main="Forecasting to June 2017",xlab="year", ylab="Unemployment Rate (%)"),
        col = c("black", "red"), lwd = 2, lty = 1:2)
legend("top", lty = c(1,3), legend = c("actual","Arima (1,1,2)"), col = c("black", "red"))

abline(v=2021)
abline(v=2016)
abline(v=2017.4)
```



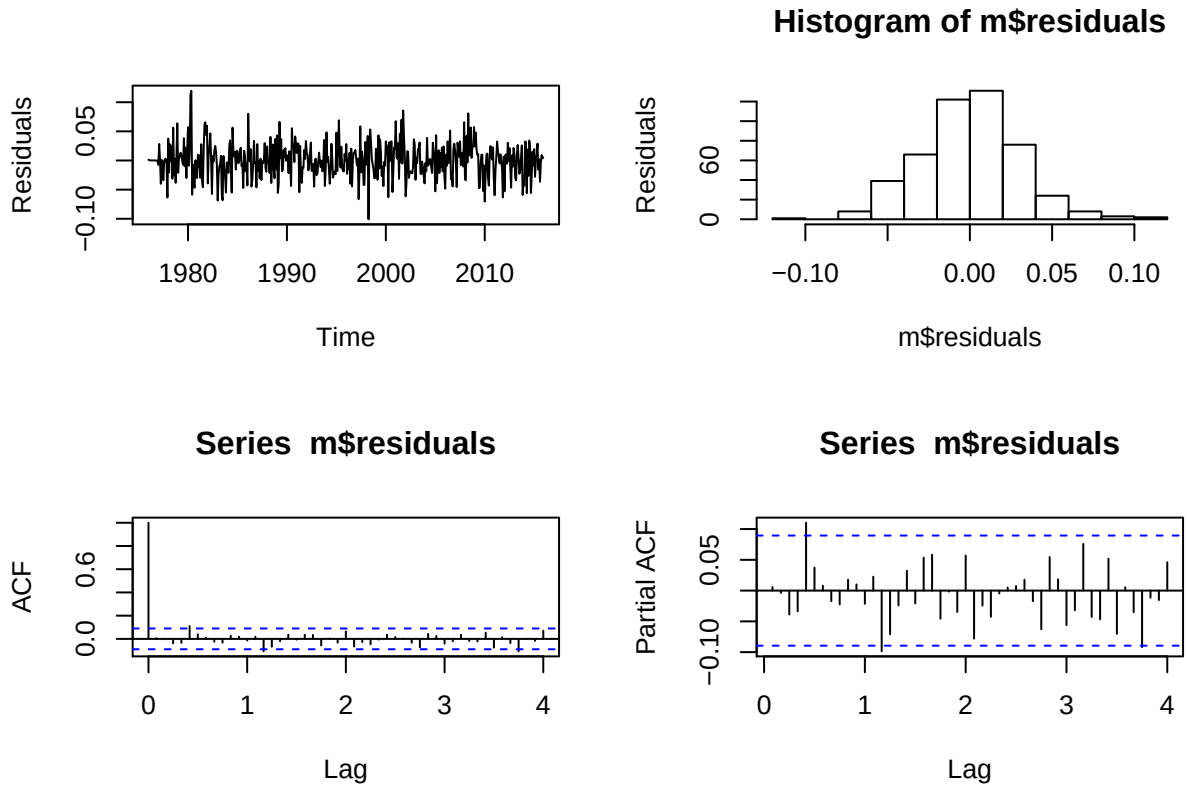

```
Box.test(m1$residuals, lag = 5)
```

```
##
##  Box-Pierce test
##
```

```
## data:  m1$residuals
## X-squared = 7.0989, df = 5, p-value = 0.2134
```

We cannot reject the null hypothesis of no correlation with p-value = 0.2134 from the Box Jjung test.

```
print_resid_chart(m1)
```



The data are independently distributed (i.e. the correlations in the population from which the sample is taken are 0, so that any observed correlations in the data result from randomness of the sampling process).

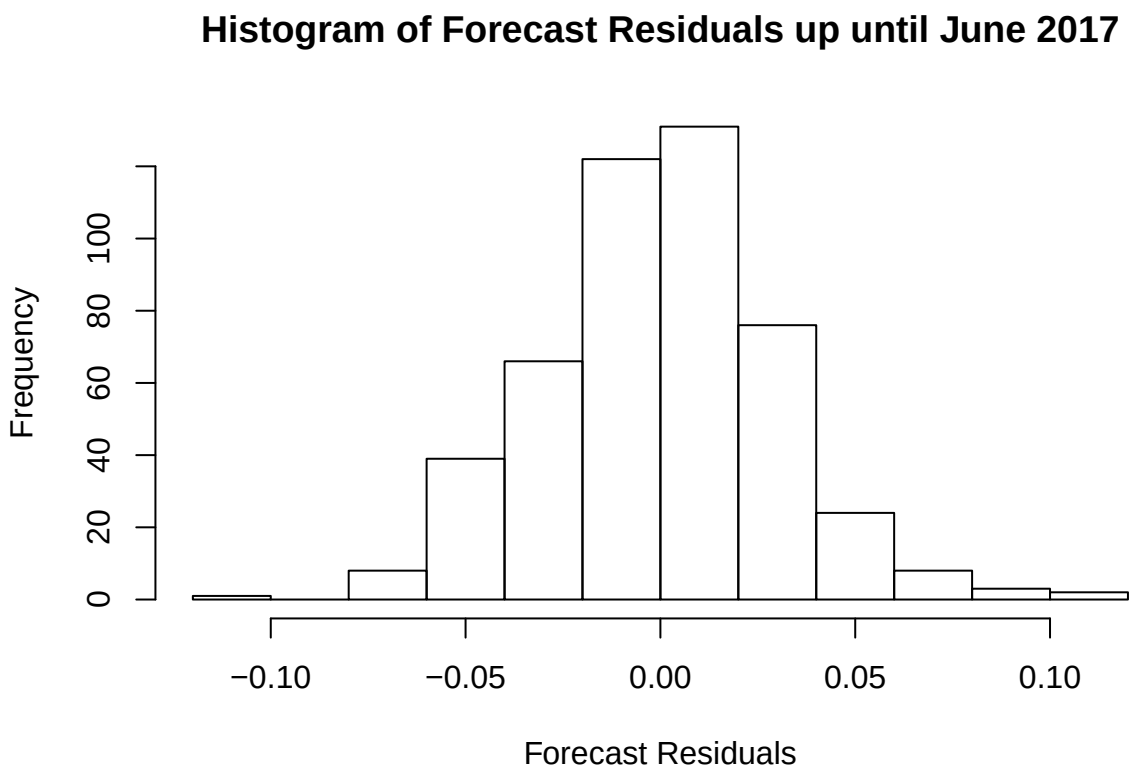
How well does your model predict the unemployment rate up until June 2017?

```
summary(m1)
```

```
## Series: emp.log
## ARIMA(1,1,2)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1     ma2    sar1     sma1
##       0.8820  -0.9835  0.2143  0.2024  -0.8779
## s.e.  0.0457   0.0623  0.0470  0.0701   0.0517
##
## sigma^2 estimated as 0.0009321:  log likelihood=962.57
## AIC=-1913.15   AICc=-1912.97   BIC=-1888.27
##
## Training set error measures:
##                       ME       RMSE        MAE        MPE     MAPE
## Training set 4.957169e-05 0.02995266 0.02305173 0.01250213 1.287594
```

```
##                      MASE        ACF1
## Training set 0.1940187 0.005626306
```

```
hist(forecast1$residuals, main="Histogram of Forecast Residuals up until June 2017",xlab="Forecast Resi
```



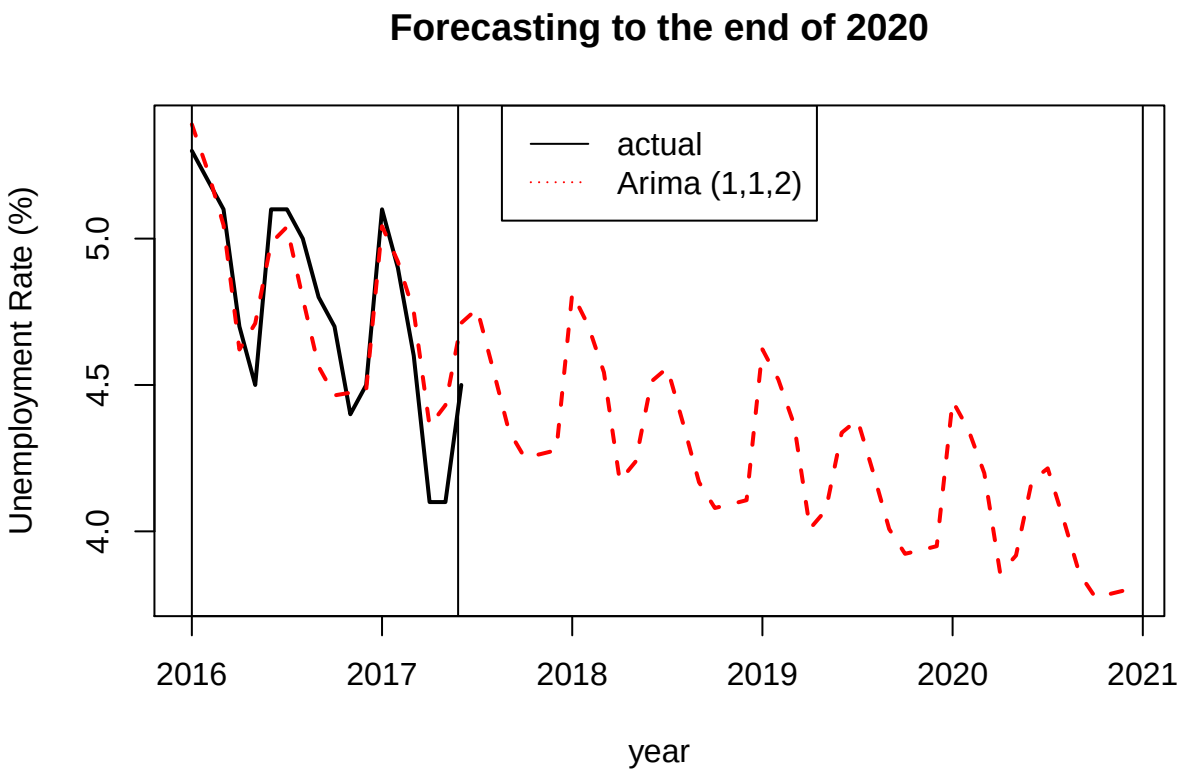

```
rmse(emp.test-exp(forecast1$mean))
```

```
## [1] 0.1644955
```

The root mean square error for the forecast to june 2017 is 0.1644955 while the true values are within 4.1 to 5.2. This is a reasonable estimation of the model.

To the end of 2020

```
m1 <- Arima(emp.log, order = c(1, 1, 2), seasonal = list(order = c(1, 1,1)))

forecast2 <- forecast(m1, h = 60)

ts.plot(emp.test,exp(forecast2$mean),
        gpars=list(main="Forecasting to the end of 2020",xlab="year", ylab="Unemployment Rate (%)"),
        col = c("black", "red"), lwd = 2, lty = 1:2)
legend("top", lty = c(1,3), legend = c("actual","Arima (1,1,2)"), col = c("black", "red"))
abline(v=2021)
abline(v=2016)
abline(v=2017.4)
```



2. What does the unemployment rate look like at the end of 2020? How credible is this estimate?

```
df_fcst <- as.data.frame(forecast2)
exp(df_fcst["Dec 2020",1])
```

```
## [1] 3.800283
```

By the end of 2020, the unemployment rate would have fallne to 3.8002826 and if the model was to extend out indefinitely, the unemployment rate would reach 0. We think using data up to Dec 2015 to predict unemployment rate up to the end of 2020, which is 6 years ahead, is not credible due to the extended period of time. We propose that a maximum of 2 years for forecasting would be best practice.

(B) Build a linear time-regression and incorporate seasonal effects. Be sure to evaluate the residuals and assess this model on the basis of the assumptions of the classical linear model, and then produce a 1 year and a 4 year forecast.

Linear Regression model

To account for seasonality in linear regression, we need to set up factors for categorical variables i.e. the months. In other words, we set up extra variables for each month and run linear regression. Fortunately, the encoding process for these variables is included in the R function "tslm" in the forecast library. We opted to use this function for brevity and readability.

```
head(time(emp.training))
```

```
##           Jan      Feb      Mar      Apr      May      Jun
## 1976 1976.000 1976.083 1976.167 1976.250 1976.333 1976.417
```

```
lm_unemp <- tslm(emp.training ~ trend + season)
summary(lm_unemp)
```

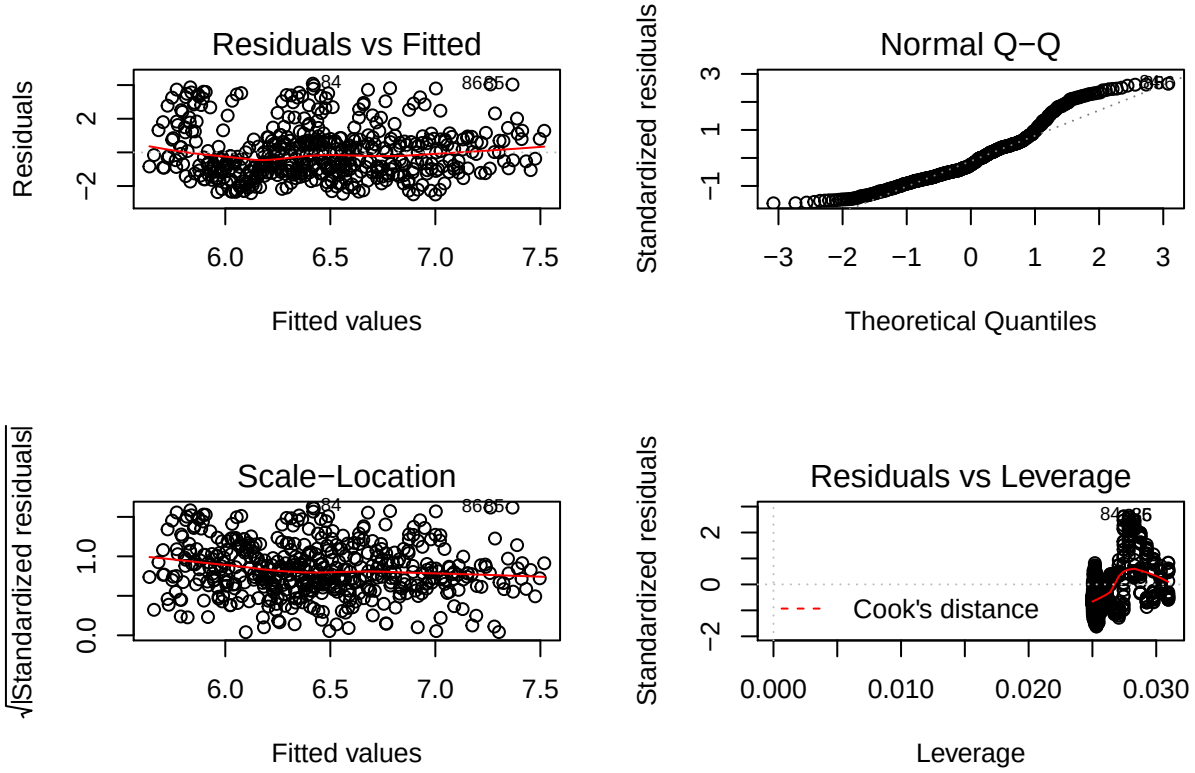
```
##
## Call:
## tslm(formula = emp.training ~ trend + season)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.500 -1.120 -0.373  0.767  4.083
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.5203359  0.2752878  27.318  < 2e-16 ***
## trend       -0.0017993  0.0005152  -3.492 0.000525 ***
## season2     -0.1032007  0.3496385  -0.295 0.768000
## season3     -0.3614014  0.3496397  -1.034 0.301840
## season4     -0.8721021  0.3496416  -2.494 0.012966 *
## season5     -0.9053028  0.3496442  -2.589 0.009920 **
## season6     -0.4310035  0.3496476  -1.233 0.218316
## season7     -0.4817042  0.3496518  -1.378 0.168966
## season8     -0.7174049  0.3496568  -2.052 0.040752 *
## season9     -0.8956056  0.3496624  -2.561 0.010740 *
## season10    -1.0213063  0.3496689  -2.921 0.003661 **
## season11    -0.9495070  0.3496761  -2.715 0.006865 **
## season12    -0.9527077  0.3496841  -2.724 0.006682 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.564 on 467 degrees of freedom
## Multiple R-squared:  0.07049,    Adjusted R-squared:  0.04661
## F-statistic: 2.951 on 12 and 467 DF,  p-value: 0.0005624
```

The values for lm_unemp are shown above with the parameters for the trend and seasons (from 2-12) with each season correspond to each month with the exception of January which is represented by the intercept. The significance for each parameter is shown.

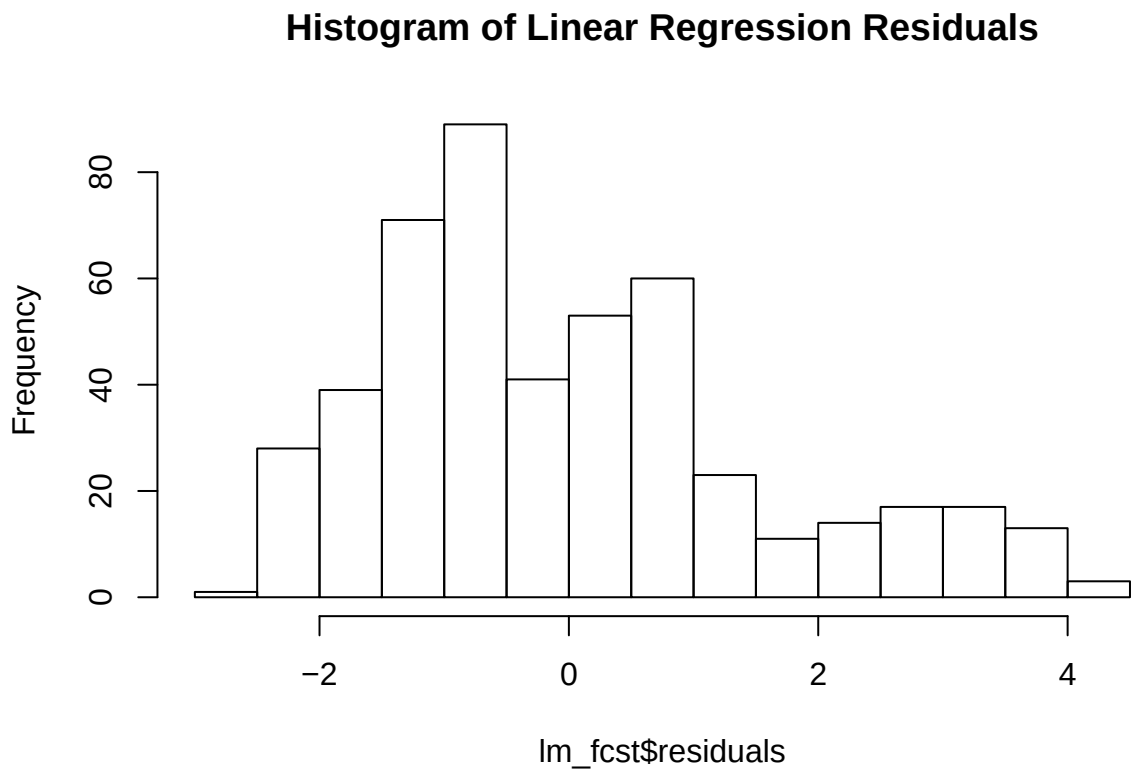
1. How well does your model predict the unemployment rate up until June 2017?

We assess the linear regression model by checking the residual plots. The residuals vs. fitted show non-zero mean; the scale-location indicates non-constant variance; the normal Q-Q plot suggests non non linearity and skewdness in the data; and the residuals vs leverage shows several points above cook's distance. All these observations together show that linear regression is not a good model, and the prediction based on this model will be biased and inaccuracte.

```
par(mfrow=c(2,2))
plot(lm_unemp)
```

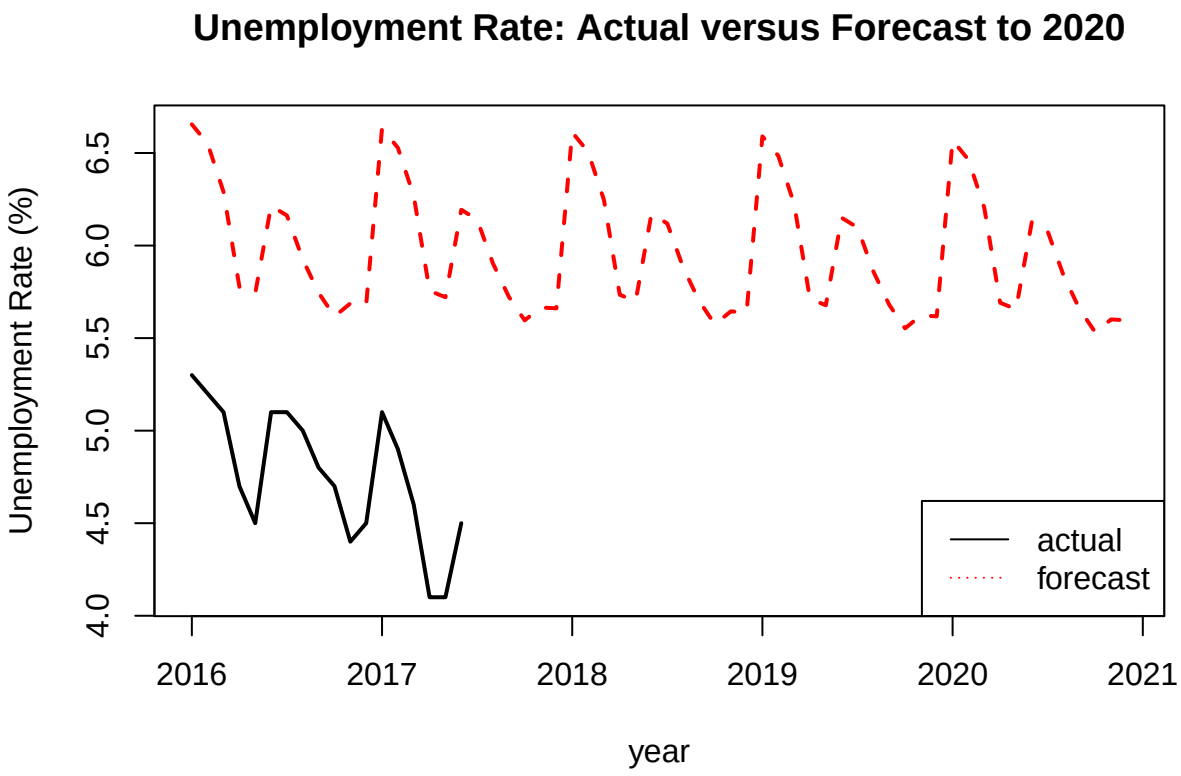



```
lm_fcst<-ts(forecast(lm_unemp, h=60))

hist(lm_fcst$residuals,main="Histogram of Linear Regression Residuals")
```



The histogram of the residuals confirm non normality of the error, violating the assumption 6 of linear regression.

```
ts.plot(emp.test,lm_fcst$mean,
        gpars=list(main="Unemployment Rate: Actual versus Forecast to 2020",xlab="year", ylab="Unemploy
        col = c("black", "red"), lwd = 2, lty = 1:2)
legend("bottomright", lty = c(1,3), legend = c("actual","forecast"), col = c("black", "red"))
```




```
rmse(emp.test-lm_fcst$mean[1:18])
```

```
## [1] 1.329487
```

The root mean square error for the prediction with the linear regression model is 1.329. This seems unacceptable given the unemployment rate is 4.1 to 5.5 range. Compared to the Arima this error is EIGHT times more.

2. What does the unemployment rate look like at the end of 2020? How credible is this estimate?

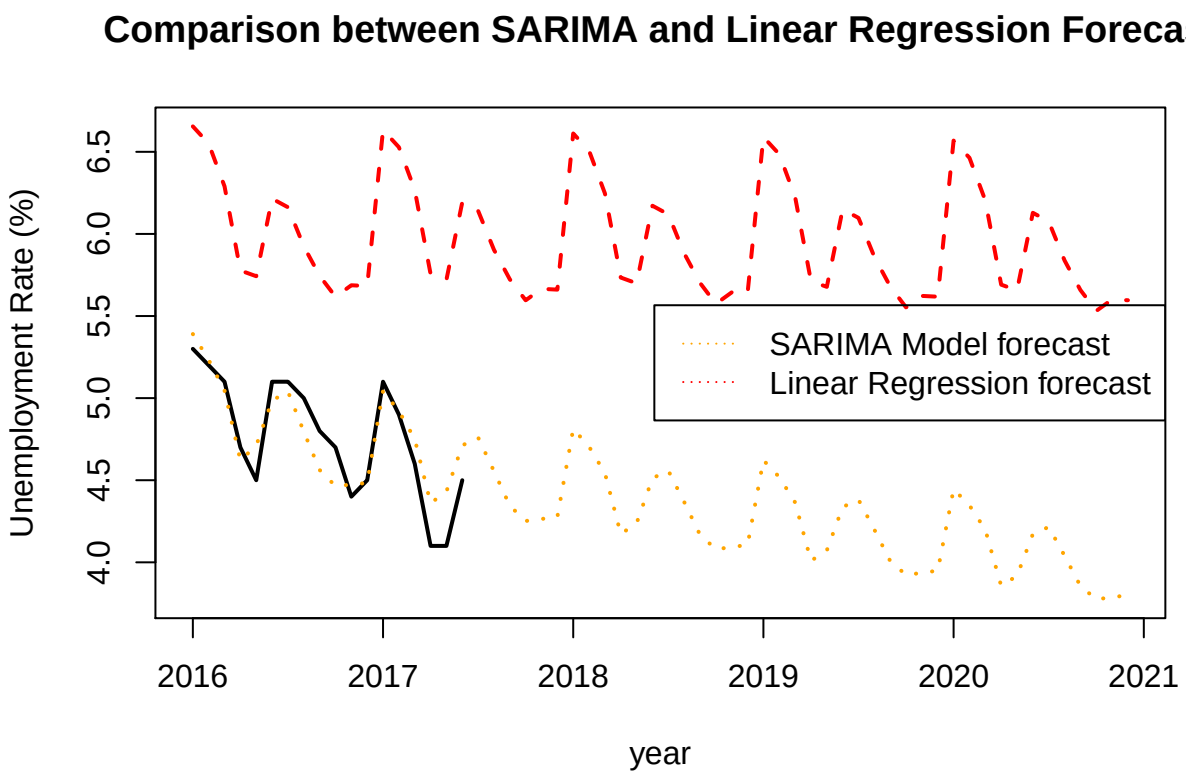
```
lm_fcst$mean[60]
```

```
##       60
## 5.596005
```

The employment rate in 2020 is 5.59 which is higher than the Arima prediction of 3.8. However, we run in the problem with extended period of prediction so this estminate is not credible. And the model is not as accurate as the Arima model in predicting the rate from 2016 to June 2017, so we think it will be even less credible than the Arima model although 5.59 is within the reasonable range of unemployment rate between 1976 and 2016.

3. Compare this forecast to the one produced by the SARIMA model. What do you notice?

```
ts.plot(emp.test,lm_fcst$mean, exp(forecast2$mean),
        gpars=list(main="Comparison between SARIMA and Linear Regression Forecast",xlab="year", ylab="U
        col = c("black", "red",'orange'), lwd = 2, lty = 1:3)
legend("right", lty = c(3,3), legend = c("SARIMA Model forecast","Linear Regression forecast")
       , col = c('orange',"red"))
```



We plot the true values in solid black, SARIMA model forecasts in dotted orange, and linear regression forecast in dotted red. Compared to the SARIMA model, the forecast from the linear regression model is higher and more conservative but doesn't follow the true values for the rate. Both models show an overall decreasing trend and seasonality; however, for the SARIMA the rate of decreasing is much higher.

Question 3: Autosale Data and VAR modeling

You also have data on automotive car sales. Use a VAR model to produce a 1 year forecast on both the unemployment rate and automotive sales for 2017 in the US. Compare the 1 year forecast for unemployment produced by the VAR and SARIMA models, examining both the accuracy AND variance of the forecast. Do you think the addition of the automotive sales data helps? Why or why not?

EDA

```
nsa <- read.csv("TOTALNSA.csv", stringsAsFactors = FALSE)
head(nsa)
```

```
##         DATE TOTALNSA
## 1 1976-01-01    885.2
## 2 1976-02-01    994.7
## 3 1976-03-01   1243.6
## 4 1976-04-01   1191.2
## 5 1976-05-01   1203.2
## 6 1976-06-01   1254.7
```

```
tail(nsa)
```

```
##           DATE TOTALNSA
## 493 2017-01-01   1164.3
## 494 2017-02-01   1352.1
## 495 2017-03-01   1582.7
## 496 2017-04-01   1449.7
## 497 2017-05-01   1544.1
## 498 2017-06-01   1500.6
```

```
describe(nsa)
```

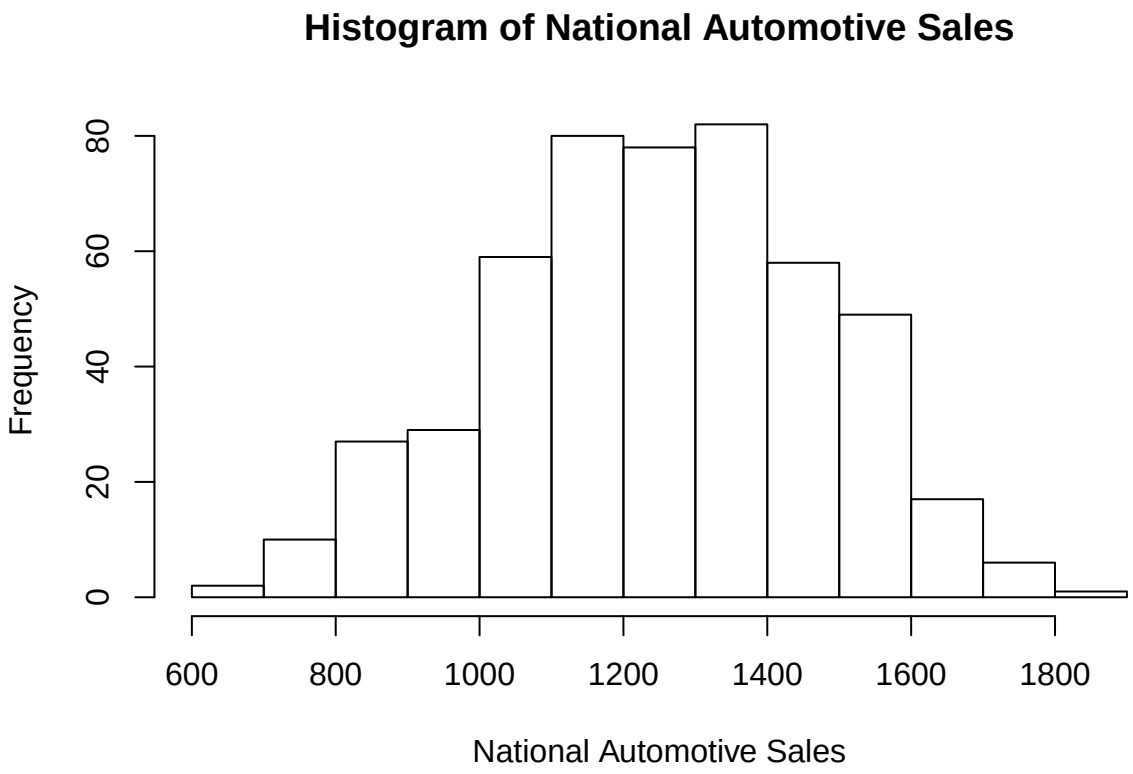
```
## Warning in describe(nsa): NAs introduced by coercion
```

```
## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning
## Inf
```

```
## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning
## -Inf
```

```
##          vars   n    mean     sd median trimmed    mad   min    max  range
## DATE*       1 498     NaN     NA     NA     NaN     NA   Inf   -Inf   -Inf
## TOTALNSA    2 498 1249.62 223.82 1253.4 1254.63 233.95 670.4 1845.7 1175.3
##           skew kurtosis    se
## DATE*       NA       NA    NA
## TOTALNSA -0.14    -0.45 10.03
```

```
totalnsa <- ts(nsa$TOTALNSA, frequency = 12, start = c(1976,1))
hist(totalnsa,main="Histogram of National Automotive Sales",xlab="National Automotive Sales")
```

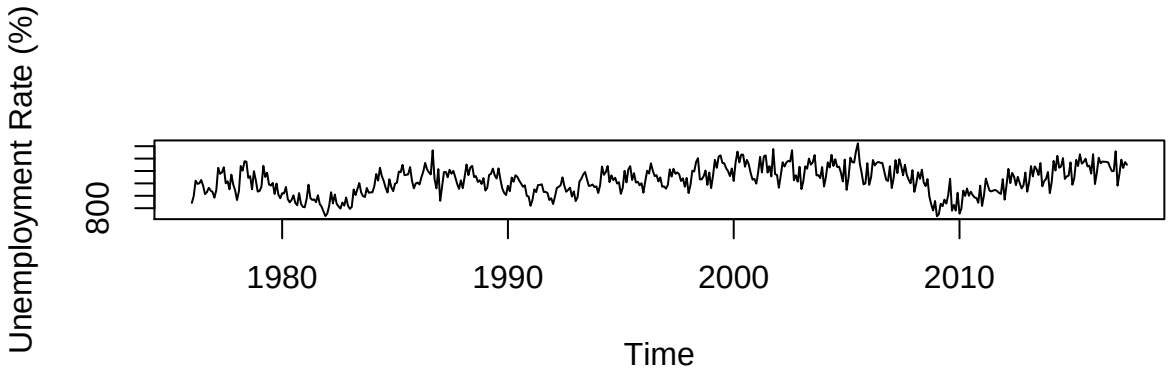


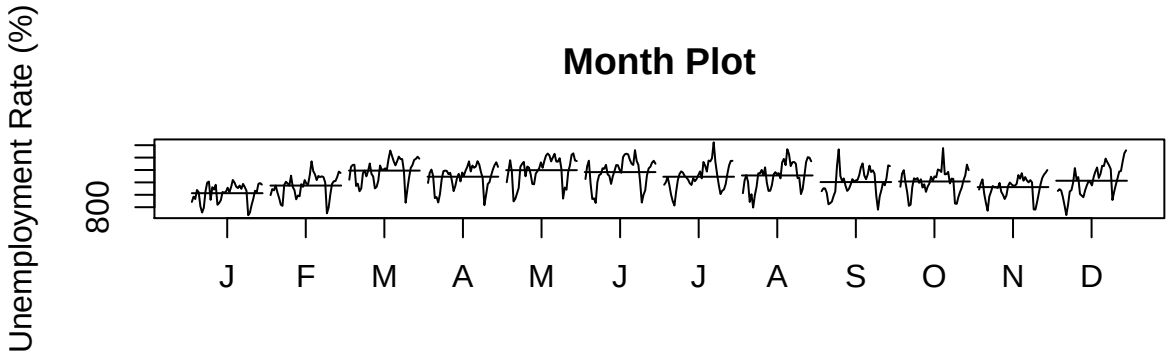

```
summary(totalnsa)
```

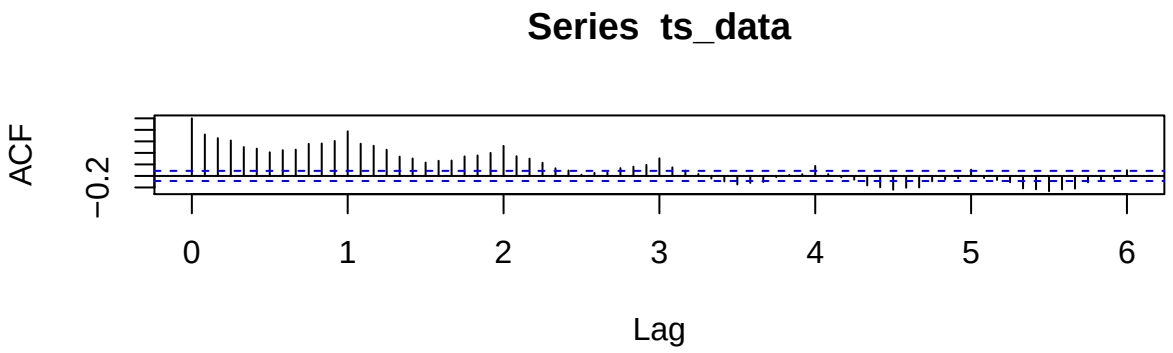
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   670.4  1094.7  1253.4  1249.6  1407.2  1845.7
```

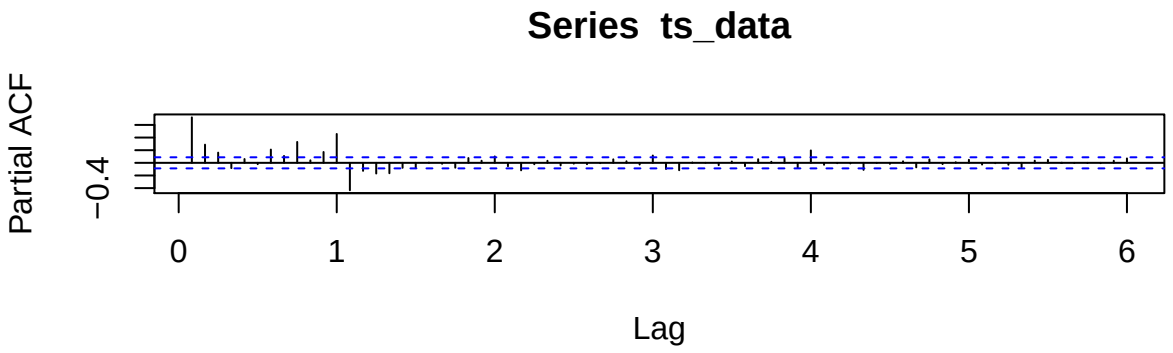
From checking the data and plotting the histogram, the autosale number seems reasonable, without any extreme outliers or extreme skewness.

```
print_tsplots(totalnsa)
```


Unemployment Rate (%)
800
1980
1990
2000
2010
Time


Month Plot
Unemployment Rate (%)
800
J
F
M
A
M
J
J
A
S
O
N
D





From the time series plots, we can see that the autosale data is not stationary. The data also exhibits seasonality from the ACF plot. And the seasonality lag seems to be at 12 month.

Modeling

The VAR modeling process begins with checking for unit root test.

```
tnsa.training <- window(totalnsa, start = c(1976,1), end = c(2015,12))
tnsa.test <- window(totalnsa, start = c(2016,1))

adf.test(emp.log)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  emp.log
## Dickey-Fuller = -2.4449, Lag order = 7, p-value = 0.3899
## alternative hypothesis: stationary
```

```
pp.test(emp.log)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  emp.log
## Dickey-Fuller Z(alpha) = -10.733, Truncation lag parameter = 5,
## p-value = 0.5107
```

```
## alternative hypothesis: stationary
```

```
adf.test(tnsa.training)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tnsa.training
## Dickey-Fuller = -3.4231, Lag order = 7, p-value = 0.04978
## alternative hypothesis: stationary
```

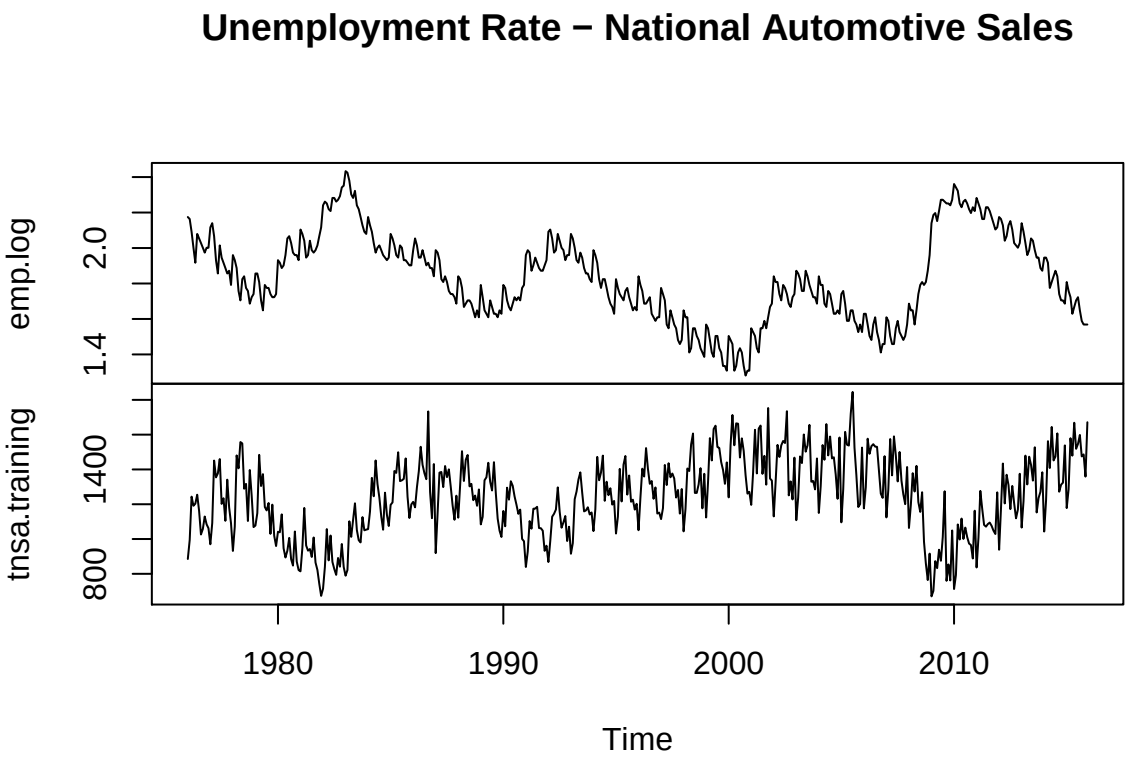
```
pp.test(tnsa.training)
```

```
## Warning in pp.test(tnsa.training): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  tnsa.training
## Dickey-Fuller Z(alpha) = -143.78, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

The adf and the unit root tests fail for the unemployment data but pass for the autosale data. So we need to check for cointegration of the two time series.

```
#combine log of unemployment data and total autosale
empnsa <- cbind(emp.log, tnsa.training)
plot.ts(empnsa, main="Unemployment Rate - National Automotive Sales")
```



From the plot, the two series do not appear to be cointegrated because they don't progress in similar fashion.

We confirm this observation with the Phillips-Ouliaris test.

```
po.test(empnsa)
```

```
## Warning in po.test(empnsa): p-value smaller than printed p-value
```

```
##
##  Phillips-Ouliaris Cointegration Test
##
## data:  empnsa
## Phillips-Ouliaris demeaned = -70.579, Truncation lag parameter =
## 4, p-value = 0.01
```

The Philips-Oularis test (p-value of 0.01) clearly rejects the hypothesis that the series are cointegrated. We can now proceed to select the order of the VAR model

Select the order of the VAR model - Use VARSelect

```
VARselect(empnsa, lag.max = 36, type = "both")
```

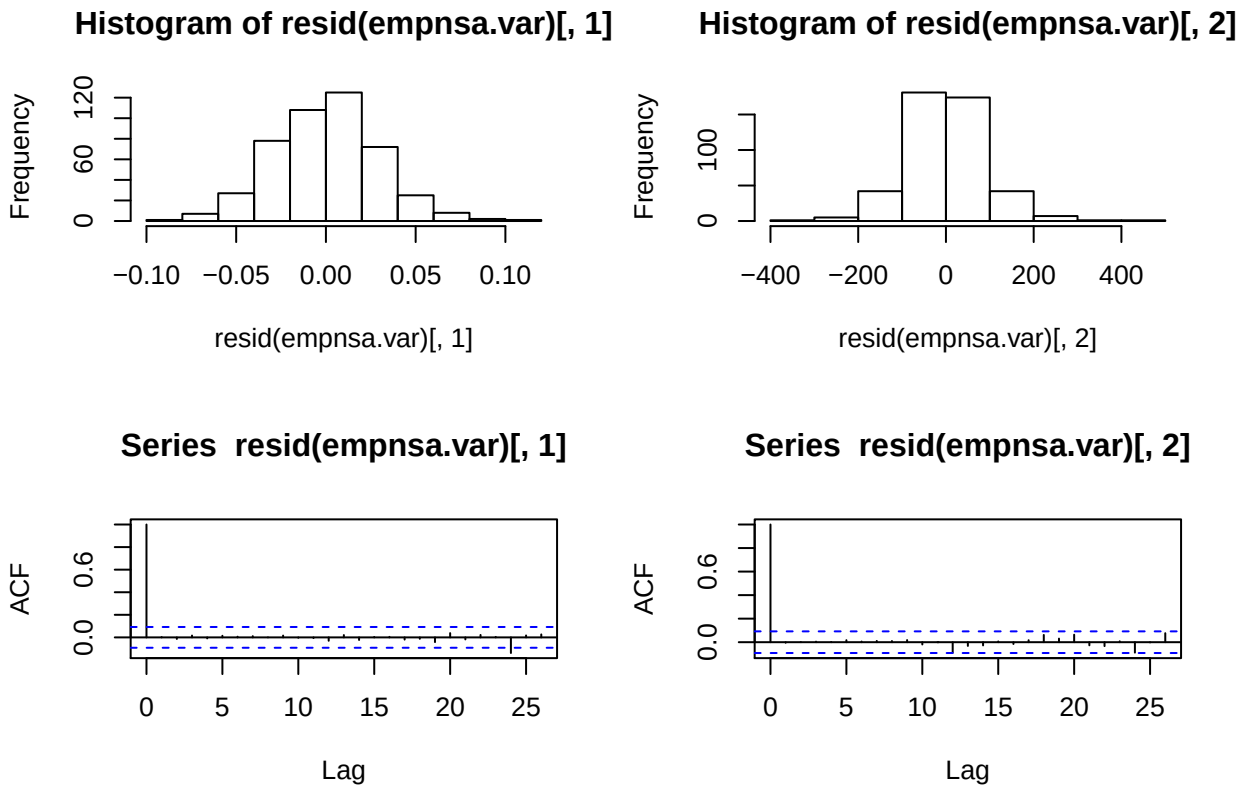
```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     26     13     13     26
##
## $criteria
##                1         2         3         4         5         6
## AIC(n)  4.310084  4.253909  4.123393  3.840790  3.812814  3.813586
## HQ(n)   4.339187  4.297563  4.181598  3.913547  3.900122  3.915446
## SC(n)   4.383883  4.364607  4.270990  4.025287  4.034210  4.071881
## FPE(n) 74.446842 70.380226 61.768926 46.562967 45.278848 45.314521
##                7         8         9        10        11        12
## AIC(n)  3.767420  3.580065  3.293044  3.189179  3.083097  2.846978
## HQ(n)   3.883831  3.711028  3.438558  3.349245  3.257714  3.036146
## SC(n)   4.062615  3.912159  3.662037  3.595072  3.525889  3.326669
## FPE(n) 43.270995 35.879055 26.927980 24.272442 21.830493 17.240239
##               13        14        15        16        17        18
## AIC(n)  2.389429  2.401435  2.391829  2.357281  2.361911  2.362279
## HQ(n)   2.593149  2.619707  2.624651  2.604655  2.623836  2.638756
## SC(n)   2.906020  2.954925  2.982218  2.984570  3.026098  3.063366
## FPE(n) 10.910925 11.043564 10.938945 10.568544 10.618780 10.624030
##               19        20        21        22        23        24
## AIC(n)  2.371296  2.378699  2.375213  2.382807  2.362235  2.338743
## HQ(n)   2.662325  2.684279  2.695344  2.717490  2.711470  2.702528
## SC(n)   3.109282  3.153585  3.186998  3.231492  3.247819  3.261226
## FPE(n) 10.721764 10.803106 10.767348 10.851460 10.632676 10.388104
##               25       26       27        28        29        30        31
## AIC(n) 2.290465 2.281992 2.294539  2.306547  2.315441  2.322610  2.335043
## HQ(n)  2.668802 2.674880 2.701979  2.728538  2.751983  2.773704  2.800689
## SC(n)  3.249847 3.278273 3.327720  3.376627  3.422420  3.466489  3.515821
## FPE(n) 9.900886 9.819909 9.946689 10.069893 10.163140 10.239811 10.371760
##               32        33        34        35        36
## AIC(n)  2.347757  2.359745  2.368566  2.350327  2.358763
## HQ(n)   2.827954  2.854493  2.877865  2.874178  2.897166
```

```
## SC(n)   3.565435  3.614322  3.660042  3.678702  3.724038
## FPE(n) 10.508619 10.639818 10.738869 10.549771 10.644474
```

We see here that the lowest AIC is for order=26

Build the VAR model using order=26

```
# build the model
empnsa.var <- VAR(empnsa, p=26, type = "trend")
#coef(empnsa.var)
#test the model
par(mfrow=c(2,2))
hist(resid(empnsa.var)[, 1])
hist(resid(empnsa.var)[, 2])
acf(resid(empnsa.var)[, 1])
acf(resid(empnsa.var)[, 2])
```

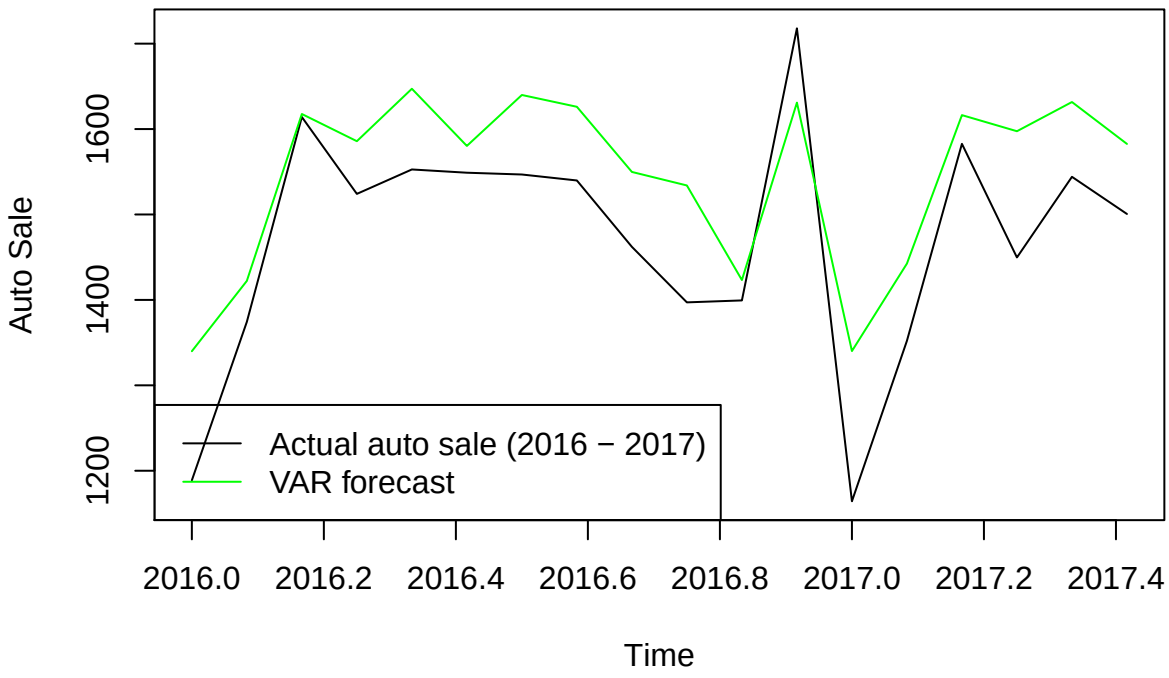


From the model diagnostic plots, we see that the VAR model with order 26 works well with normal distributions of the residuals for both time series. And the ACFs look like white noise.

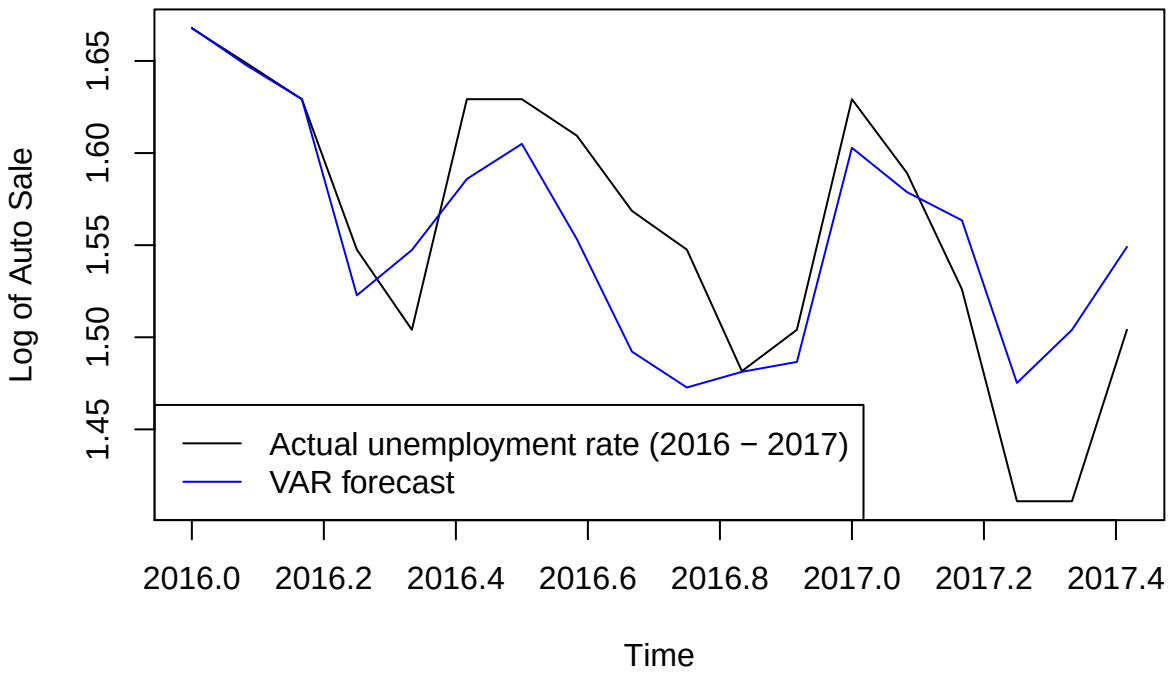
```
ccf(resid(empnsa.var)[, 1], resid(empnsa.var)[, 2])
```

the ccd plot verifies that the cross correlations are approx 0 for all non zero lags so the residuals are bivariate white noise.

```
# make prediction
empnsa.pred <- predict(empnsa.var, n.ahead = 18)

#Extract data from the prediction
emp.pred <- ts(empnsa.pred$fcst$emp.log[,1], st = c(2016,1), fr=12)
tnsa.pred <- ts(empnsa.pred$fcst$tnsa.training[,1], st = c(2016,1), fr=12)
ts.plot(tnsa.test, type = 'l',ylab="Auto Sale")
lines(tnsa.pred, type = 'l', col = 'green')
legend("bottomleft", lty = c(1,1), legend = c("Actual auto sale (2016 - 2017)","VAR forecast"), col = c
```

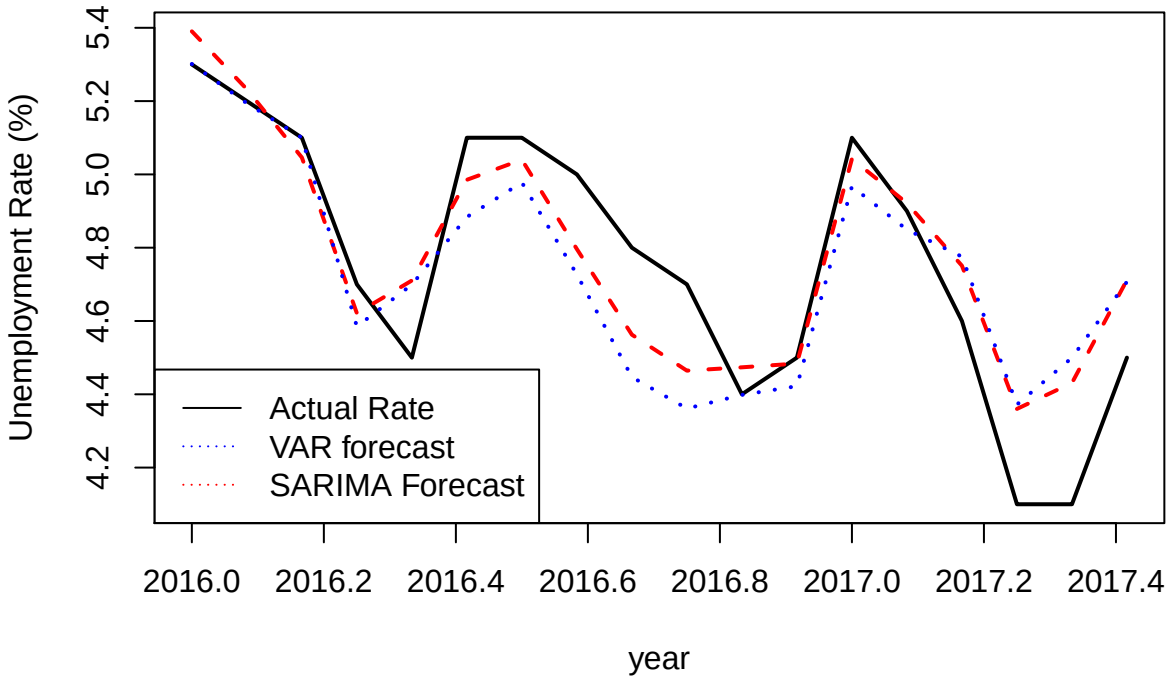



```
ts.plot(emp.test.log, type = 'l',ylab="Log of Auto Sale")
lines(emp.pred, type = 'l', col = 'blue')
legend("bottomleft", lty = c(1,1), legend = c("Actual unemployment rate (2016 - 2017)","VAR forecast"),
```



Compare the 1 year forecast for unemployment produced by the VAR and SARIMA models, examining both the accuracy AND variance of the forecast. Do you think the addition of the automotive sales data helps? Why or why not?

```
ts.plot(emp.test, window(exp(forecast1$mean), end = c(2017,6)), exp(emp.pred),
        gpars=list(xlab="year", ylab="Unemployment Rate (%)"),
        col = c("black", "red", "blue"), lwd = 2, lty = 1:3)
legend("bottomleft", lty = c(1,3,3), legend = c("Actual Rate","VAR forecast","SARIMA Forecast"), col =
```



The black curve indicate true Values whereas the dotted lines are the ones based on the model. Dotted Blue is for the VAR and red is for the SARIMA.

```
print('rmse and variance for SARIMA forecast')
```

```
## [1] "rmse and variance for SARIMA forecast"
```

```
rmse(emp.test - exp(forecast1$mean))
```

```
## [1] 0.1644955
```

```
var(emp.test - exp(forecast1$mean))
```

```
## [1] 0.02830649
```

```
print('rmse and variance for VAR forecast')
```

```
## [1] "rmse and variance for VAR forecast"
```

```
rmse(emp.test - exp(emp.pred))
```

```
## [1] 0.2058122
```

```
var(emp.test - exp(emp.pred))
```

```
## [1] 0.04423921
```

```
summary(exp(emp.pred))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.361   4.460   4.717   4.737   4.946   5.301
```

Both the root mean square error and the variance of the SARIMA model forecast are lower than those obtained from the VAR model. Therefore, adding the autosale model does not help with the forecasting of the unemployment rate. For the VAR model to work better than the SARIMA, the added variables should have some sort of influence over the unemployment rate. Automotive sale is unlikely to influence the broad employment outside of the automotive industry. It can be argued also that the unemployment rate might have more influence on autosale, hence the relationship between auto sale and and unemployment rate might not be equal, which is a prerequisite for the VAR model's performance.