Lab 3

Sue Yang, Michelle Kim, Legg Yeung

August 6, 2017

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?

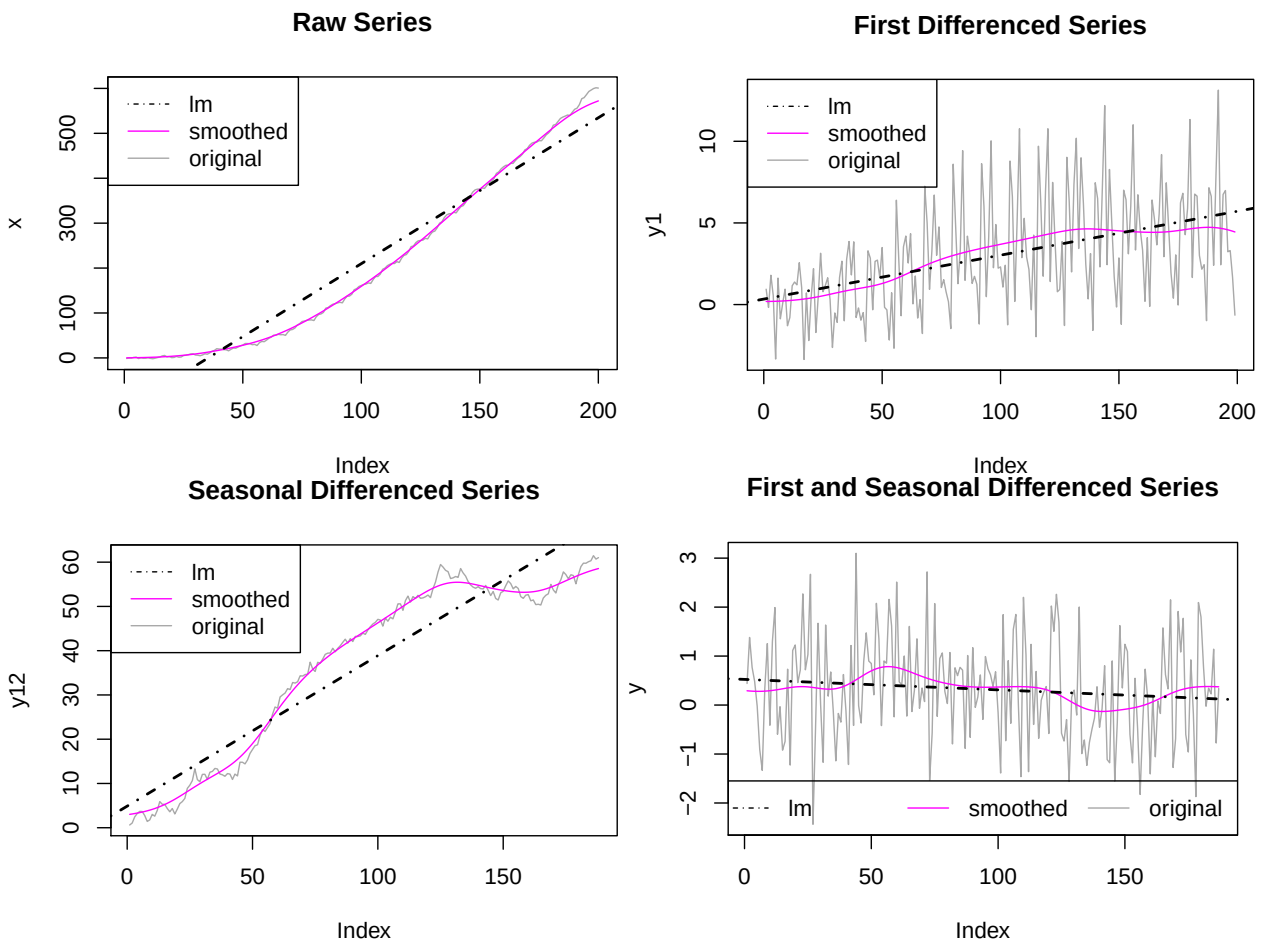
To remove linear trend it is necessary to difference once with lag 1. Because in a purely linear trend, differencing by the last observation singles out the roughly equivalent increments which are stationary at the mean. To remove seasonal patterns, it is necessary to difference once with lag 12 for annual patterns, or lag 4 for quarterly patterns. Because in a purely seasonal series where each observation is same as the one from the last season, differencing by the observation from last season removes seasonal fluctuations and stablize the series at the mean. Real world economic time series, like ours, often exhibit both linear trend in additional to annual seasonal behavior. Therefore differencing by lag 1 and lag 12 will help stablize the raw series at the mean. In the following we simulate a raw series that is a random walk with drift (linear trend) and strong dependence on its seasonal lag 1, then demonstrate how first and seasonal differencing can stablize it at the mean. The model is :

$$(1-B)(1-B^{12})X_t = 0.5 + W_t$$

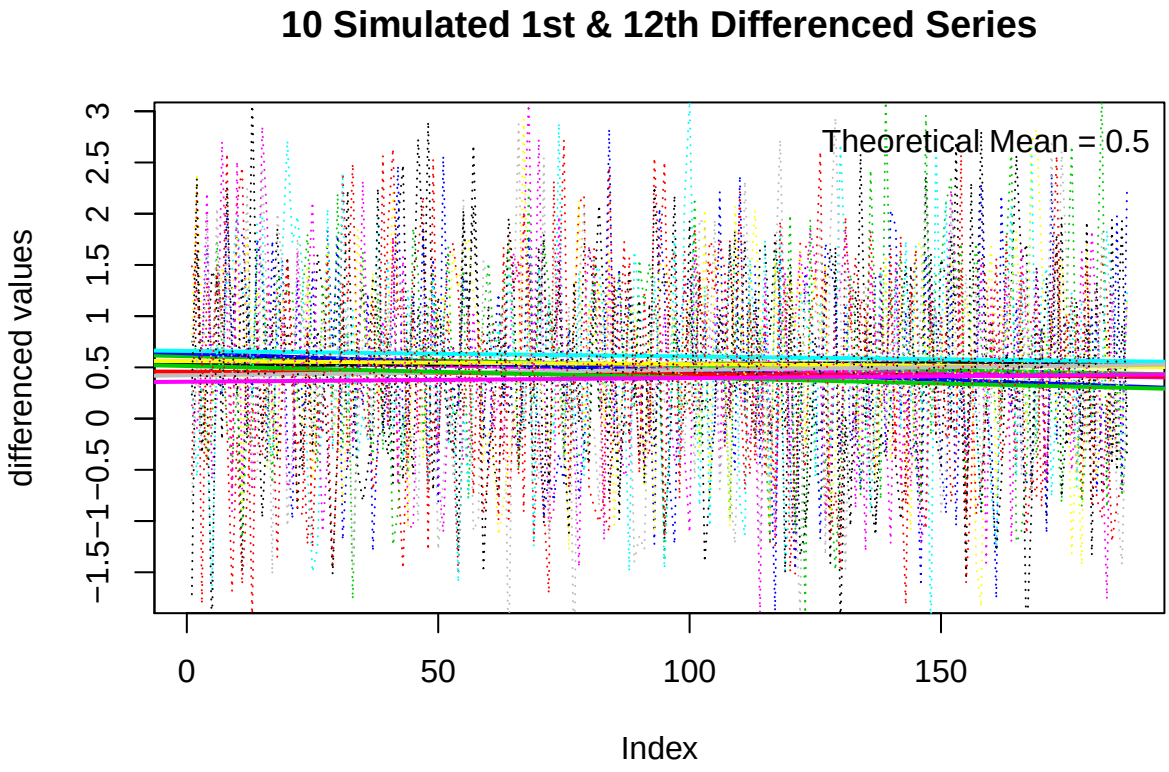
$$X_t = 0.5 + X_{t-12} - X_{t-13} + X_{t-1} + w_t$$

```
set.seed(30)
x<-w<-rnorm(200)

#x_t = 0.5 + x_{t-12}-*x_{t-13}+x_{i-1}+w_t
for(i in 14:200) x[i]<- 0.5 + x[i-12]- x[i-13]+x[i-1]+w[i]


#difference only once
y1<-diff(x)

#difference only seasonally
y12<-diff(x,12)

#difference lag 1 and seasonally
y<-diff(diff(x),12)

# Kernel smoothing
x.k.smooth.widest = ksmooth(time(x),
                            x, kernel = c("normal"),
                            bandwidth = 25)
y1.k.smooth.widest = ksmooth(time(y1),
                            y1, kernel = c("normal"),
                            bandwidth = 25)
y12.k.smooth.widest = ksmooth(time(y12),
```

```
                              y12, kernel = c("normal"),
                              bandwidth = 25)
y.k.smooth.widest = ksmooth(time(y),
                              y, kernel = c("normal"),
                              bandwidth = 25)

# Make plots

plot(x, type="l", col = "darkgray")
title("Raw Series")
lines(x.k.smooth.widest$x, x.k.smooth.widest$y, col = "magenta")
abline(lm(x~time(x)), lty = "dotdash", col = "black", lwd = 2)
legend("topleft", legend = c("lm", "smoothed", "original"),
       lty = c("dotdash","solid", "solid"),
       col = c("black", "magenta", "darkgrey"))

plot(y1, type="l", col = "darkgray")
title("First Differenced Series")
lines(y1.k.smooth.widest$x, y1.k.smooth.widest$y, col = "magenta")
abline(lm(y1~time(y1)), lty = "dotdash", col = "black", lwd =2)
legend("topleft", legend = c("lm", "smoothed", "original"),
       lty = c("dotdash","solid", "solid"),
       col = c("black", "magenta", "darkgrey"))

plot(y12, type="l", col = "darkgray")
title("Seasonal Differenced Series")
lines(y12.k.smooth.widest$x, y12.k.smooth.widest$y, col = "magenta")
abline(lm(y12~time(y12)), lty = "dotdash", col = "black", lwd = 2)
legend("topleft",
       legend = c("lm", "smoothed", "original"),
       lty = c("dotdash","solid", "solid"),
       col = c("black", "magenta", "darkgrey"))

plot(y, col = "darkgray", type = "l")
title("First and Seasonal Differenced Series")
lines(y.k.smooth.widest$x, y.k.smooth.widest$y, col = "magenta")
abline(lm(y~time(y)), lty = "dotdash", col = "black", lwd =2)
legend("bottom", legend = c("lm", "smoothed", "original"),
       lty = c("dotdash","solid", "solid"),
       col = c("black", "magenta", "darkgrey"),
       horiz = TRUE)
```




```
y = list()
for (j in 1:10){
  set.seed(j)
  x<-w<-rnorm(200)
  for(i in 14:200) x[i]<- 0.5 + x[i-12]- x[i-13]+x[i-1]+w[i]
  y[[j]]<-diff(diff(x),12)
}

#par(mfrow = c(2,2))

plot(y[[1]], col = rgb(0,0,0, alpha = 0), ylab = "differenced values",
     main = "10 Simulated 1st & 12th Differenced Series")
axis(side = 2, at = seq(-2,3,0.5))
for (j in 1:10){
lines(y[[j]], col = j, type = "l", lwd = 1, lty = "dotted")
abline(lm(y[[j]]~time(y[[1]])),
       col = j*3, lwd = 2)
}
legend("topright",legend = "Theoretical Mean = 0.5", bty = "n")
```



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 administrations 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.

```
rm(list = ls())
library(moments)
library(psych)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.3.2
```

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 3.3.2
```

```
library(effects)
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.3.2
```

```
library(sandwich)
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2015). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2. http://CRAN.R-project.org/package=stargazer
```

```
library(vars)
```

```
## Loading required package: MASS
## Warning: package 'MASS' was built under R version 3.3.2
## Loading required package: strucchange
## Loading required package: zoo
## Warning: package 'zoo' was built under R version 3.3.2
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
## Loading required package: urca
## Loading required package: lmtest
## Warning: package 'lmtest' was built under R version 3.3.2
```

```
opts_chunk$set(tidy.opts=list(width.cutoff=55),tidy=TRUE)

unem.data = read.csv("UNRATENSA.csv", header = T)
auto.data = read.csv("TOTALNSA.csv", header = T)

unem.ts = ts(unem.data$UNRATENSA, frequency = 12, start = c(1948,1))
auto.ts = ts(auto.data$TOTALNSA, frequency = 12, start = c(1976,1))
```

Exploratory Data Analysis

Data Overview

In order to choose the best form for our models, we will first conduct a thorough EDA of the available datasets. We begin by examining the structure of the data.

```
str(unem.data)
```

```
## 'data.frame':    834 obs. of  2 variables:
##  $ DATE     : Factor w/ 834 levels "1948-01-01","1948-02-01",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ UNRATENSA: num  4 4.7 4.5 4 3.4 3.9 3.9 3.6 3.4 2.9 ...
```

```
str(auto.data)
```

```
## 'data.frame':    498 obs. of  2 variables:
##  $ DATE    : Factor w/ 498 levels "1976-01-01","1976-02-01",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ TOTALNSA: num  885 995 1244 1191 1203 ...
```

We note that the unemployment rate data begins in January 1948, whereas the auto sales data begins in January 1976.

```
cbind(head(unem.data), tail(unem.data))
```

```
##         DATE UNRATENSA       DATE UNRATENSA
## 1 1948-01-01       4.0 2017-01-01       5.1
## 2 1948-02-01       4.7 2017-02-01       4.9
## 3 1948-03-01       4.5 2017-03-01       4.6
## 4 1948-04-01       4.0 2017-04-01       4.1
## 5 1948-05-01       3.4 2017-05-01       4.1
## 6 1948-06-01       3.9 2017-06-01       4.5
```

```
cbind(head(auto.data), tail(auto.data))
```

```
##         DATE TOTALNSA       DATE TOTALNSA
## 1 1976-01-01    885.2 2017-01-01   1164.3
## 2 1976-02-01    994.7 2017-02-01   1352.1
## 3 1976-03-01   1243.6 2017-03-01   1582.7
## 4 1976-04-01   1191.2 2017-04-01   1449.7
## 5 1976-05-01   1203.2 2017-05-01   1544.1
## 6 1976-06-01   1254.7 2017-06-01   1500.6
```

Both datasets end in June 2017.

```
nrow(unem.data) - nrow(auto.data)
```

```
## [1] 336
```

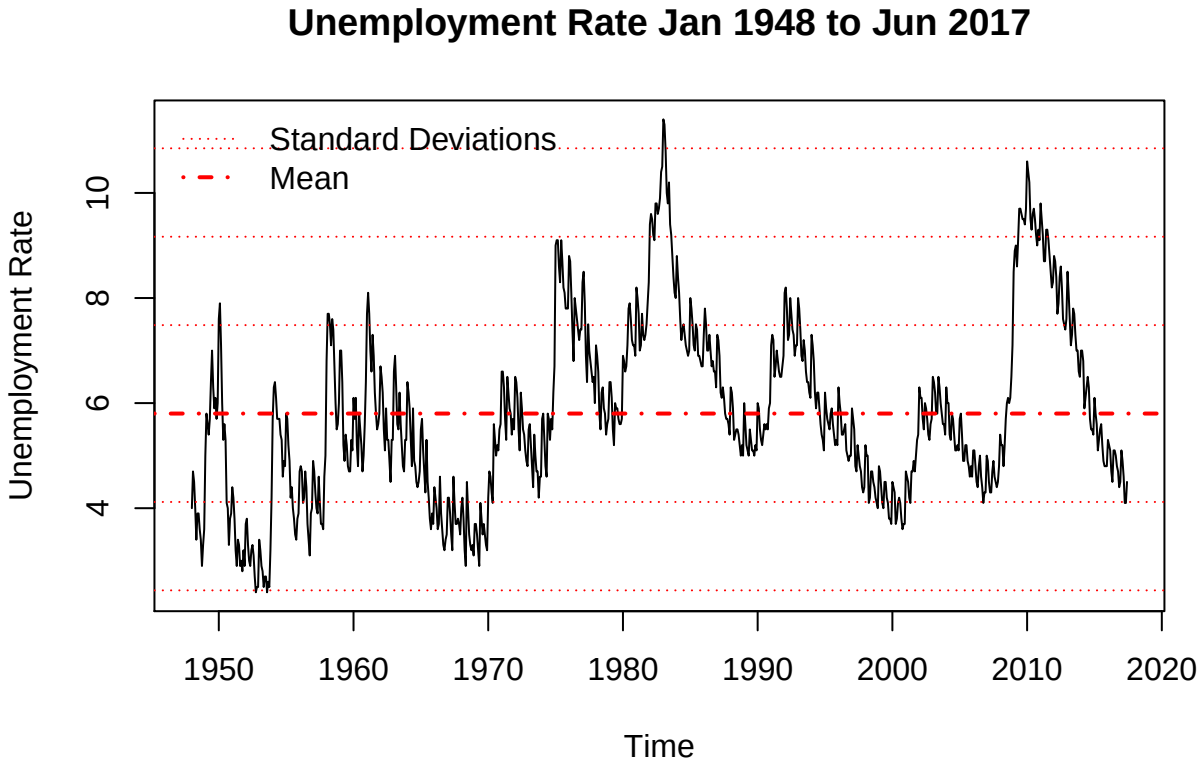
Both datasets are time indexed, accompanied with a key variable of interests. With UNRATENSA, the key variable refers to unemployment rate. With TOTALNSA, the key variable refers to car sale. Both time series present monthly data. UNRATENSA has 28 more years of data, which is 336 more observations than TOTALNSA.

Time Series Overview

Time Plots and Histograms

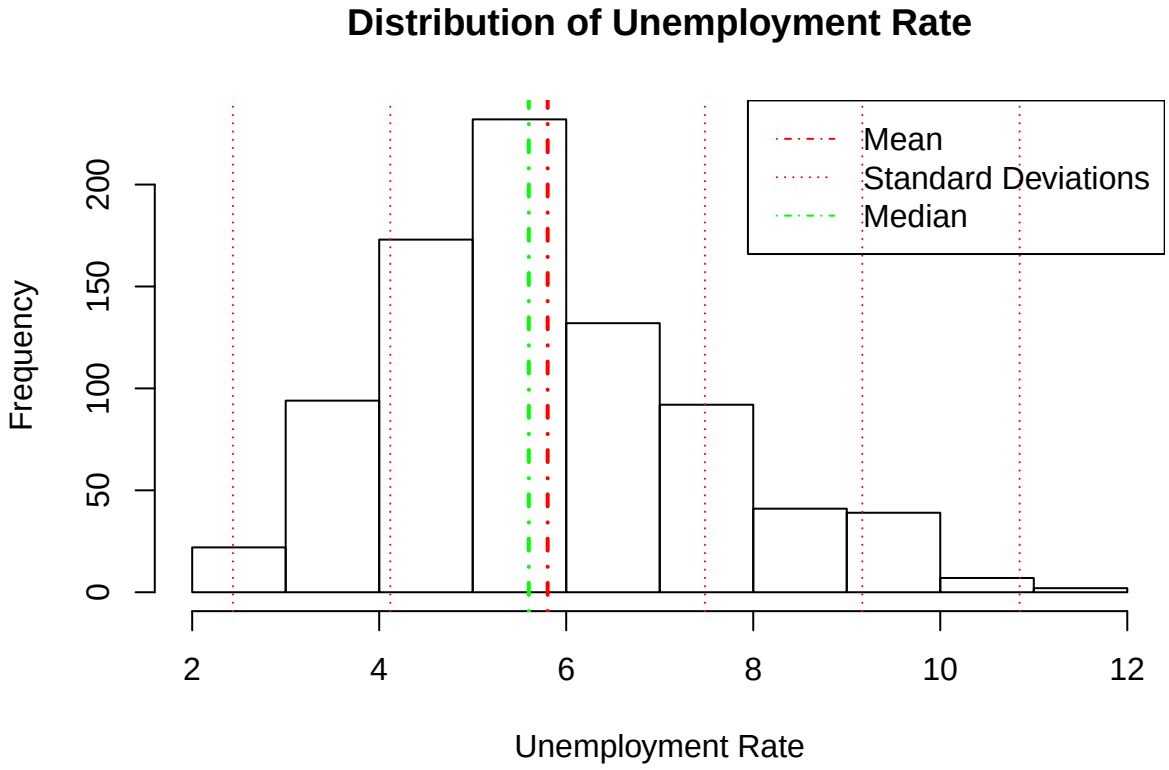
Next we will examine a plot of the time series of the Unemployment Rate data.

```
ts.plot(unem.ts, ylab = "Unemployment Rate")
title("Unemployment Rate Jan 1948 to Jun 2017")
abline(h = mean(unem.ts), col = "red", lty = "dotdash",
    lwd = 2)
abline(h = c((mean(unem.ts) + sd(unem.ts)), (mean(unem.ts) +
    2 * sd(unem.ts)), (mean(unem.ts) + 3 * sd(unem.ts))),
    col = "red", lty = "dotted", lwd = 1)
abline(h = c((mean(unem.ts) - sd(unem.ts)), (mean(unem.ts) -
    2 * sd(unem.ts))), col = "red", lty = "dotted", lwd = 1)

# abline(lm(unem.ts~time(unem.ts)), lty = 'dotted', col
# = 'blue')
legend("topleft", c("Standard Deviations", "Mean"), col = c("red",
    "red"), lty = c("dotted", "dotdash"), bty = "n", lwd = c(1,
    2))
```



From the above time plot of unemployment, we see clear persistency of the observations. That is, when observations are above or below the mean, they tend to stay so for a while. The overall trend seem to climb slowly upward.

```
hist(unem.data$UNRATENSA, main = "Distribution of Unemployment Rate",
    xlab = "Unemployment Rate")

abline(v = mean(unem.data$UNRATENSA), col = "red", lty = "dotdash",
    lwd = 2)
abline(v = c((mean(unem.data$UNRATENSA) + sd(unem.data$UNRATENSA)),
    (mean(unem.data$UNRATENSA) + 2 * sd(unem.data$UNRATENSA)),
    (mean(unem.data$UNRATENSA) + 3 * sd(unem.data$UNRATENSA))),
    col = "red", lty = "dotted", lwd = 1)
abline(v = c((mean(unem.data$UNRATENSA) - sd(unem.data$UNRATENSA)),
    (mean(unem.data$UNRATENSA) - 2 * sd(unem.data$UNRATENSA))),
    col = "red", lty = "dotted", lwd = 1)

abline(v = median(unem.data$UNRATENSA), col = "green", lty = "dotdash",
    lwd = 2)

legend("topright", c("Mean", "Standard Deviations", "Median"),
    col = c("red", "red", "green"), lty = c("dotdash", "dotted",
        "dotdash"))
```



Almost 5% of the observations lie 2 standard deviations above the mean, which is made obvious in the right skewed histogram as well. Next we will examine the more extreme values (more than 2.5 standard deviations from the mean.)

```
unem.data[unem.data$UNRATENSA > mean(unem.data$UNRATENSA) +
    2.5 * sd(unem.data$UNRATENSA), ]
```

```
##           DATE UNRATENSA
## 419 1982-11-01      10.4
## 420 1982-12-01      10.5
## 421 1983-01-01      11.4
## 422 1983-02-01      11.3
## 423 1983-03-01      10.8
## 426 1983-06-01      10.2
## 745 2010-01-01      10.6
## 746 2010-02-01      10.4
## 747 2010-03-01      10.2
```

```
# percentage of observations beyond 2, 2.5 and 3 sd
nrow(unem.data[unem.data$UNRATENSA > mean(unem.data$UNRATENSA) +
    2 * sd(unem.data$UNRATENSA), ])/nrow(unem.data)
```

```
## [1] 0.04916067
```

```
nrow(unem.data[unem.data$UNRATENSA > mean(unem.data$UNRATENSA) +
    2.5 * sd(unem.data$UNRATENSA), ])/nrow(unem.data)
```

```
## [1] 0.01079137
```

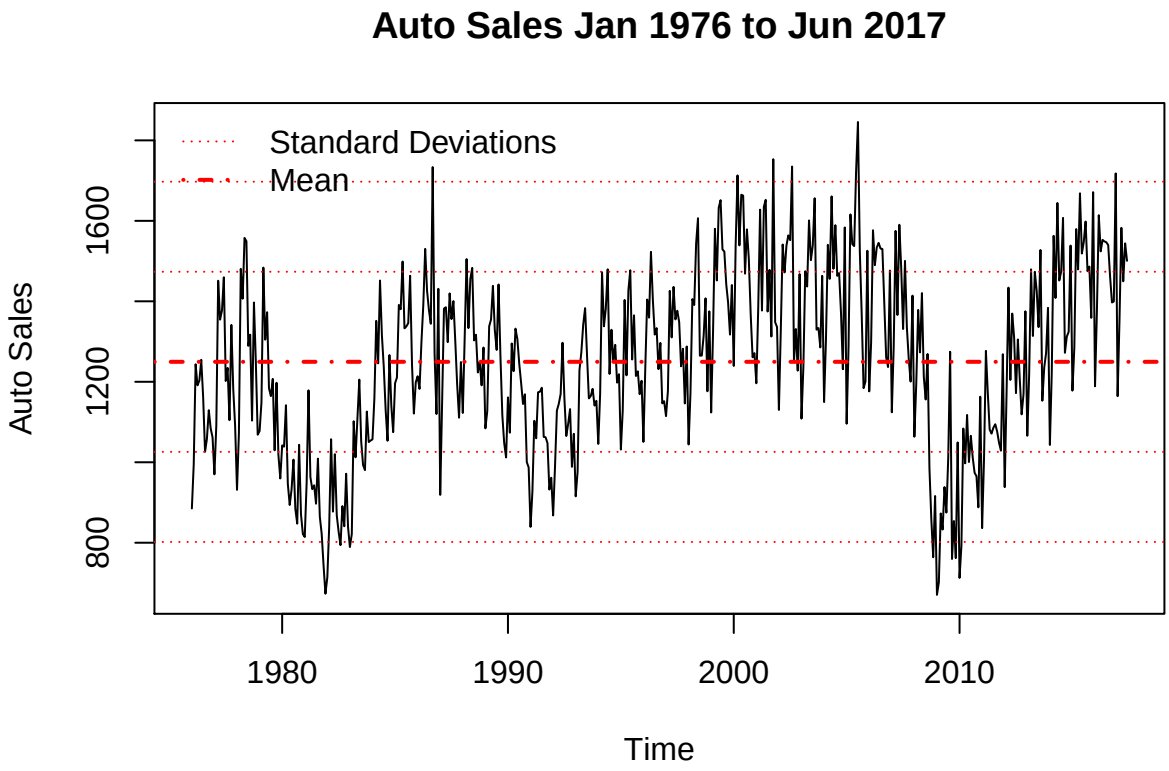
```
nrow(unem.data[unem.data$UNRATENSA > mean(unem.data$UNRATENSA) +
    3 * sd(unem.data$UNRATENSA), ])/nrow(unem.data)
```

```
## [1] 0.002398082
```

Isolating these observations, the 6 in 1982 and 1983 probably correspond to the early 1980s recessions which officially ended in November 1982. The 3 in 2010 likely correspond to the late 2000s recession which officially ended in June 2009.

Next we will examine a plot of the time series of the Auto Sales data.

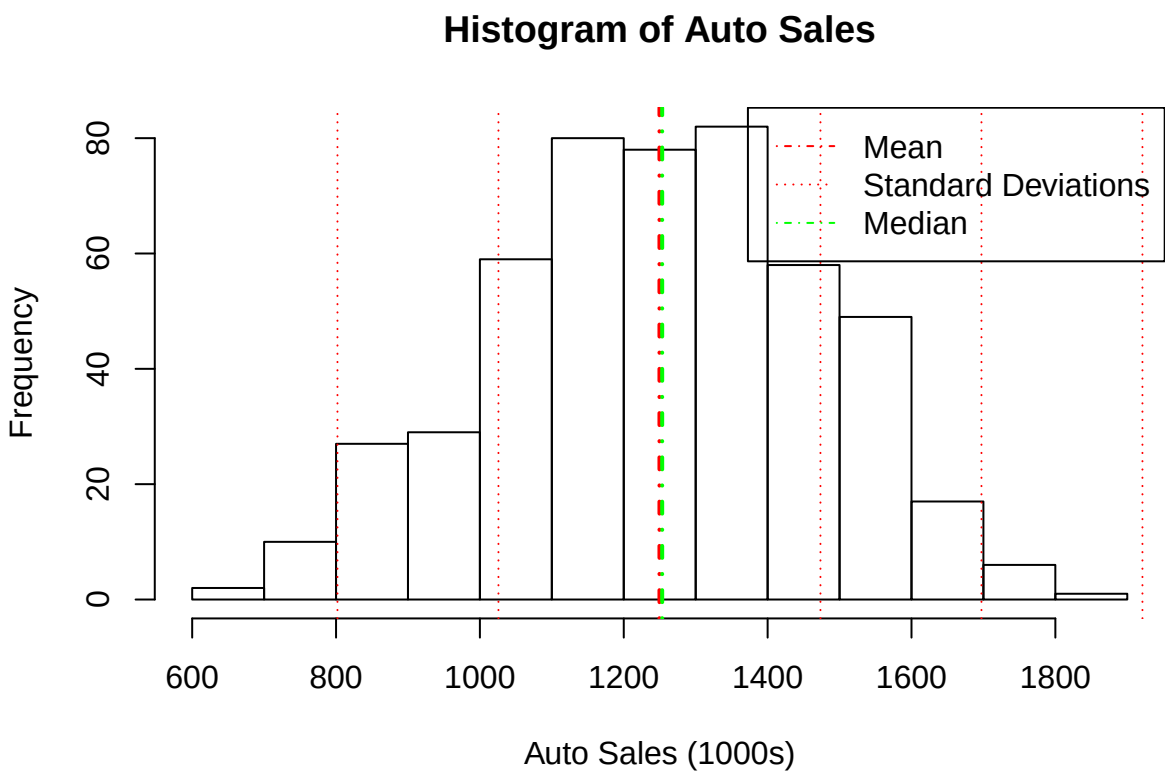
```
ts.plot(auto.ts, ylab = "Auto Sales")
title("Auto Sales Jan 1976 to Jun 2017")
abline(h = mean(auto.ts), col = "red", lty = "dotdash",
    lwd = 2)
abline(h = c((mean(auto.ts) + sd(auto.ts)), (mean(auto.ts) +
    2 * sd(auto.ts)), (mean(auto.ts) + 3 * sd(auto.ts))),
    col = "red", lty = "dotted", lwd = 1)
abline(h = c((mean(auto.ts) - sd(auto.ts)), (mean(auto.ts) -
    2 * sd(auto.ts))), col = "red", lty = "dotted", lwd = 1)

# abline(lm(auto.ts~time(auto.ts)), lty = 'dotted', col
# = 'blue')
legend("topleft", c("Standard Deviations", "Mean"), col = c("red",
    "red"), lty = c("dotted", "dotdash"), bty = "n", lwd = c(1,
    2))
```



From the above time plot of auto sales, we see some persistency, but it appears weaker than in the unemployment series. The overall trend doesn't seem to climb upward or downward. There are more noticeable seasonal patterns than in the unemployment series.

```
hist(auto.data$TOTALNSA, main = "Histogram of Auto Sales",
    xlab = "Auto Sales (1000s)")

abline(v = mean(auto.data$TOTALNSA), col = "red", lty = "dotdash",
    lwd = 2)
```

```
abline(v = c((mean(auto.data$TOTALNSA) + sd(auto.data$TOTALNSA)),
    (mean(auto.data$TOTALNSA) + 2 * sd(auto.data$TOTALNSA)),
    (mean(auto.data$TOTALNSA) + 3 * sd(auto.data$TOTALNSA))),
    col = "red", lty = "dotted", lwd = 1)
abline(v = c((mean(auto.data$TOTALNSA) - sd(auto.data$TOTALNSA)),
    (mean(auto.data$TOTALNSA) - 2 * sd(auto.data$TOTALNSA))),
    col = "red", lty = "dotted", lwd = 1)

abline(v = median(auto.data$TOTALNSA), col = "green", lty = "dotdash",
    lwd = 2)

legend("topright", c("Mean", "Standard Deviations", "Median"),
    col = c("red", "red", "green"), lty = c("dotdash", "dotted",
        "dotdash"))
```



The histogram is more symmetric and normal. Less than 1.5% of the observations lie more than 2 standard deviations above the mean and less than 2.5% of the observations lie more than 2 standard deviations below the mean. We will isolate and further examine these more extreme observations.

```
auto.data[auto.data$TOTALNSA < mean(auto.data$TOTALNSA) -
    2 * sd(auto.data$TOTALNSA), ]
```

```
##           DATE TOTALNSA
## 71  1981-11-01    743.0
## 72  1981-12-01    673.2
## 73  1982-01-01    714.4
## 80  1982-08-01    794.0
## 85  1983-01-01    789.3
## 395 2008-11-01    763.9
## 397 2009-01-01    670.4
```

```
## 398 2009-02-01    701.7
## 405 2009-09-01    759.6
## 407 2009-11-01    761.8
## 409 2010-01-01    712.5
## 410 2010-02-01    793.3
```

```
head(auto.data[auto.data$TOTALNSA > mean(auto.data$TOTALNSA) +
    2 * sd(auto.data$TOTALNSA), ], 1)
```

```
##           DATE TOTALNSA
## 129 1986-09-01   1733.6
```

```
# percentage of observations beyond 2 sd
nrow(auto.data[auto.data$TOTALNSA > mean(auto.data$TOTALNSA) +
    2 * sd(auto.data$TOTALNSA), ])/nrow(auto.data)
```

```
## [1] 0.01405622
```

```
nrow(auto.data[auto.data$TOTALNSA < mean(auto.data$TOTALNSA) -
    2 * sd(auto.data$TOTALNSA), ])/nrow(auto.data)
```

```
## [1] 0.02409639
```

After isolating these observations we note the 5 in 1982 and 1983 and the 5 in 2008 and 2009 probably correspond to the two aforementioned recessions. Notice that these recession-related observations in auto sales tend to happen a few months before those in unemployment. There is an unusual spike in 1986, possibly attributed to oil prices dropping in half that year.

```
summary(unem.data$UNRATENSA)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.400   4.700   5.600   5.801   6.900  11.400
```

```
summary(auto.data$TOTALNSA)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   670.4  1095.0  1253.0  1250.0  1407.0  1846.0
```

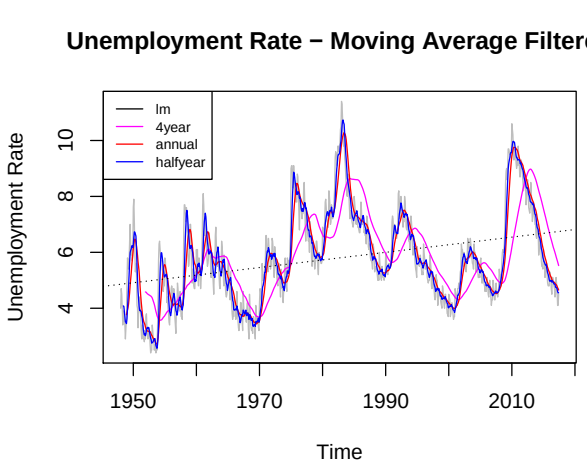
Examining the summary statistics of each series, we note that the Unemployment Rate falls between 2.4% and 11.4%, with a mean of 5.8%. The Auto Sales ranges from 670,400 cars to 1.8 million cars, with a mean of 1.2 million.

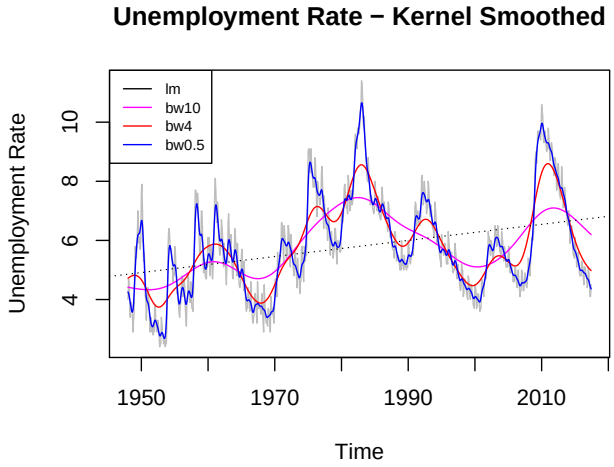
Trend Examination with Smoothers and Decomposition - Unemployment Rate

In this section, we apply smoothers and decomposition to gain some initial insights about the overall trend, seasonality and noise behavior of the unemployment rate series.

```
# Moving Average Filter
unem.ma.smooth.4year = filter(unem.ts, sides = 1, rep(1/48,
    48))
unem.ma.smooth.annual = filter(unem.ts, sides = 1, rep(1/12,
    12))
unem.ma.smooth.halfyear = filter(unem.ts, sides = 1, rep(1/6,
    6))

# Make plot
plot(unem.ts, col = "gray", ylab = "Unemployment Rate",
    main = "Unemployment Rate - Moving Average Filtered")
lines(unem.ma.smooth.4year, col = "magenta")
```

```
lines(unem.ma.smooth.annual, col = "red")
lines(unem.ma.smooth.halfyear, col = "blue")
abline(lm(unem.ts ~ time(unem.ts)), lty = "dotted", col = "black")
legend("topleft", legend = c("lm", "4year", "annual", "halfyear"),
    lty = c("solid", "solid", "solid", "solid"), col = c("black",
        "magenta", "red", "blue"), pt.cex = 1, cex = 0.7)

# Kernel smoothing
unem.k.smooth.widest = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 10)

unem.k.smooth.wide = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 4)

unem.k.smooth.narrow = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 0.5)

# Make plot
plot(unem.ts, col = "gray", ylab = "Unemployment Rate",
    main = "Unemployment Rate - Kernel Smoothed")
lines(unem.k.smooth.widest$x, unem.k.smooth.widest$y, col = "magenta")
lines(unem.k.smooth.wide$x, unem.k.smooth.wide$y, col = "red")
lines(unem.k.smooth.narrow$x, unem.k.smooth.narrow$y, col = "blue")
abline(lm(unem.ts ~ time(unem.ts)), lty = "dotted", col = "black")
legend("topleft", legend = c("lm", "bw10", "bw4", "bw0.5"),
    lty = c("solid", "solid", "solid", "solid"), col = c("black",
        "magenta", "red", "blue"), pt.cex = 1, cex = 0.7)
```





The asymmetric moving average filters a moving average over the past 6 months, 12 months and 4 years. The symmetric kernel smoothers attempted three different bandwidths (10, 4, and 0.5). In either case, the smoothed series resemble behavior of random walks with drift, evidenced by the gradually widening variance and slowly increasing mean towards the right. Formulation for the two smoothers are given here:

- One-sided Moving Average Smoother

$$m_t = \frac{1}{n+1} \sum_{j=0}^{n} x_{t-j}$$

where m_t refers to the trend we hope to study, x_t is the raw series, n is the number of past months to include

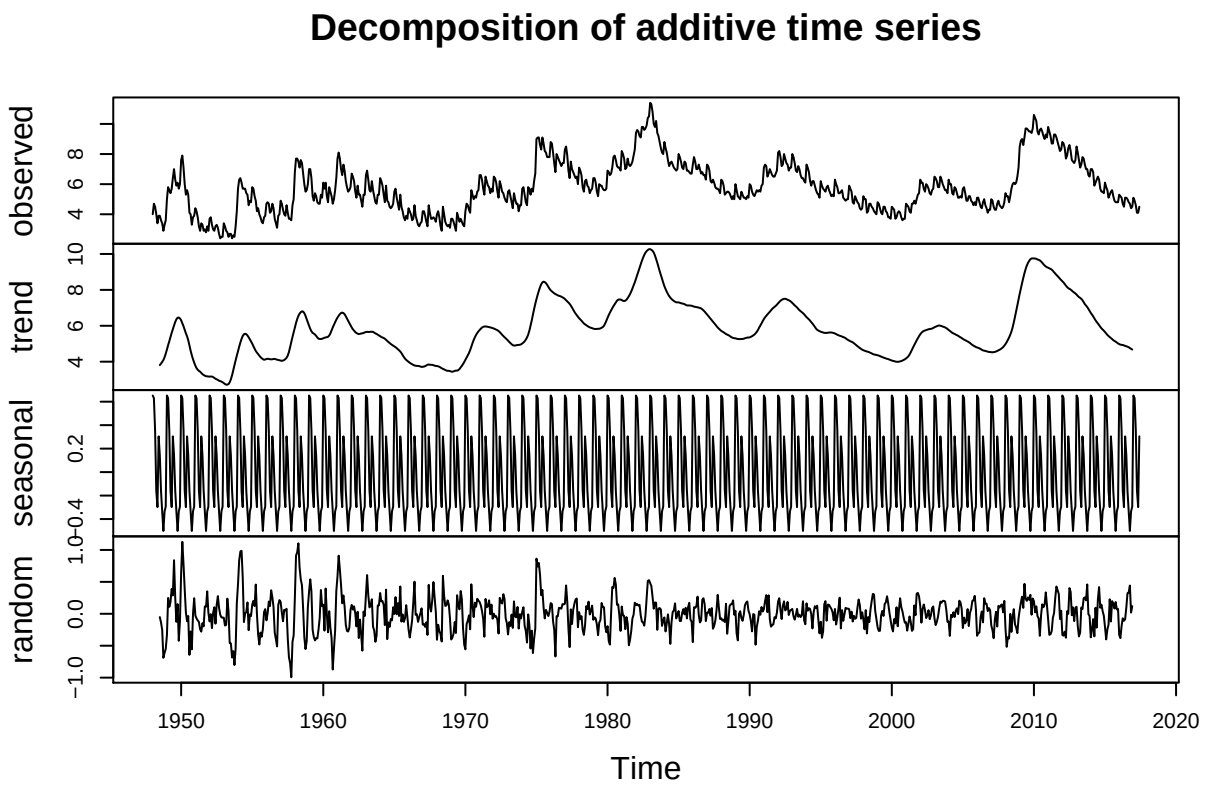
in the averaging.

- Symmertic Kernel Smoother (symmetric moving average smoother with probability density weight function applied)

$$m_t = \sum_{i=1}^{n} w_i(t) x_{t_i}$$

where density weight function $w_i(t)$ is a function of the kernel function and $w_i(t) = K(\frac{t-t_i}{b}) / \sum_{j=1}^{n} K(\frac{t-t_j}{b})$. b is the bandwidth which we manipulate. The blue kernel curve used $b = 0.5$ to correspond approximately smoothing over about half a year. The smoother curves were generated using higher bandwidth.

```
plot(decompose(unem.ts, type = "additive"))
```



The decomposed trend series resembles that of the moving average and kernel filters. Here the growing variance of the trend is more noticeable, as well as the gradual upward trend. A regular, annual seasonality series was isolated out from the raw series. The random component series is clearly non-stationary. Its variance diminishes over time, with dramatic spikes before 1965. Clearly, an OLS model would not be the right choice, and we should not disregard the non-stationary trend and random components.

These techniques assume that the raw series is composed of a clear trend, seasonality and random component. The model imposed is:

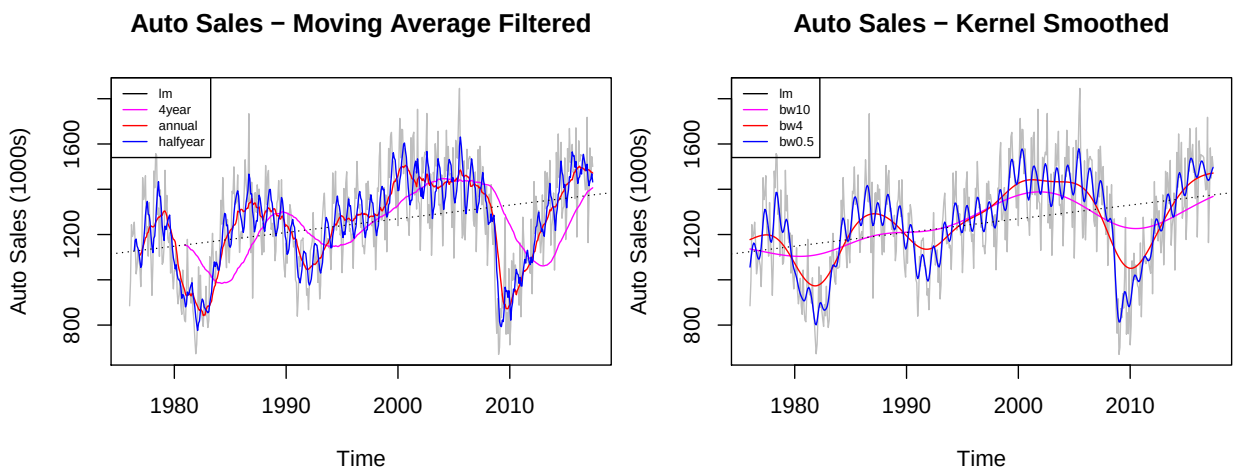
$$X_t = M_t + S_t + N_t$$

where X_t is the raw series, M_t is the trend, S_t is the seasonal component and N_t is the random component.

Trend Examination with Smoothers and Decomposition - Auto Sales

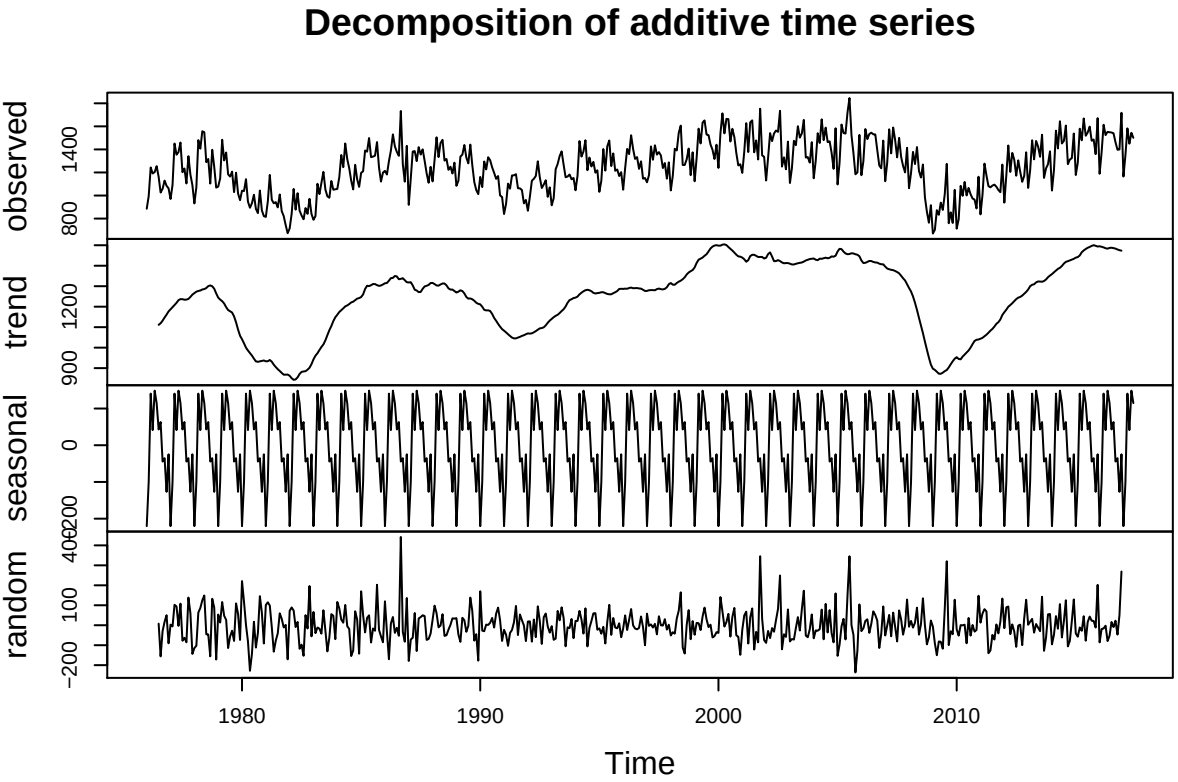
Next we will conduct a similar analysis for the auto sales data.

```
# Moving Average Filter
auto.ma.smooth.4year = filter(auto.ts, sides = 1, rep(1/60,
    60))
auto.ma.smooth.annual = filter(auto.ts, sides = 1, rep(1/12,
    12))
auto.ma.smooth.halfyear = filter(auto.ts, sides = 1, rep(1/6,
    6))

# Make plot
plot(auto.ts, col = "gray", ylab = "Auto Sales (1000s)",
    main = "Auto Sales - Moving Average Filtered")
lines(auto.ma.smooth.4year, col = "magenta")
lines(auto.ma.smooth.annual, col = "red")
lines(auto.ma.smooth.halfyear, col = "blue")
abline(lm(auto.ts ~ time(auto.ts)), lty = "dotted", col = "black")
legend("topleft", legend = c("lm", "4year", "annual", "halfyear"),
    lty = c("solid", "solid", "solid", "solid"), col = c("black",
        "magenta", "red", "blue"), pt.cex = 1, cex = 0.6)


# Kernel smoothing
auto.k.smooth.widest = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 10)

auto.k.smooth.wide = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 4)

auto.k.smooth.narrow = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 0.5)

# Make plot
plot(auto.ts, col = "gray", ylab = "Auto Sales (1000s)",
    main = "Auto Sales - Kernel Smoothed")
lines(auto.k.smooth.widest$x, auto.k.smooth.widest$y, col = "magenta")
lines(auto.k.smooth.wide$x, auto.k.smooth.wide$y, col = "red")
lines(auto.k.smooth.narrow$x, auto.k.smooth.narrow$y, col = "blue")
abline(lm(auto.ts ~ time(auto.ts)), lty = "dotted", col = "black")
legend("topleft", legend = c("lm", "bw10", "bw4", "bw0.5"),
    lty = c("solid", "solid", "solid", "solid"), col = c("black",
        "magenta", "red", "blue"), pt.cex = 1, cex = 0.6)
```



Compared to the unemployment series, the auto sales series exhibits less downhill-uphill-downhill behavior, and appears to be relatively stable between the two aforementioned recessions in early 1980s and late 2000s. It resembles less of a random walk and drift is not entirely apparent. The seasonal pattern is also stronger.

```
plot(decompose(auto.ts, type = "additive"))
```

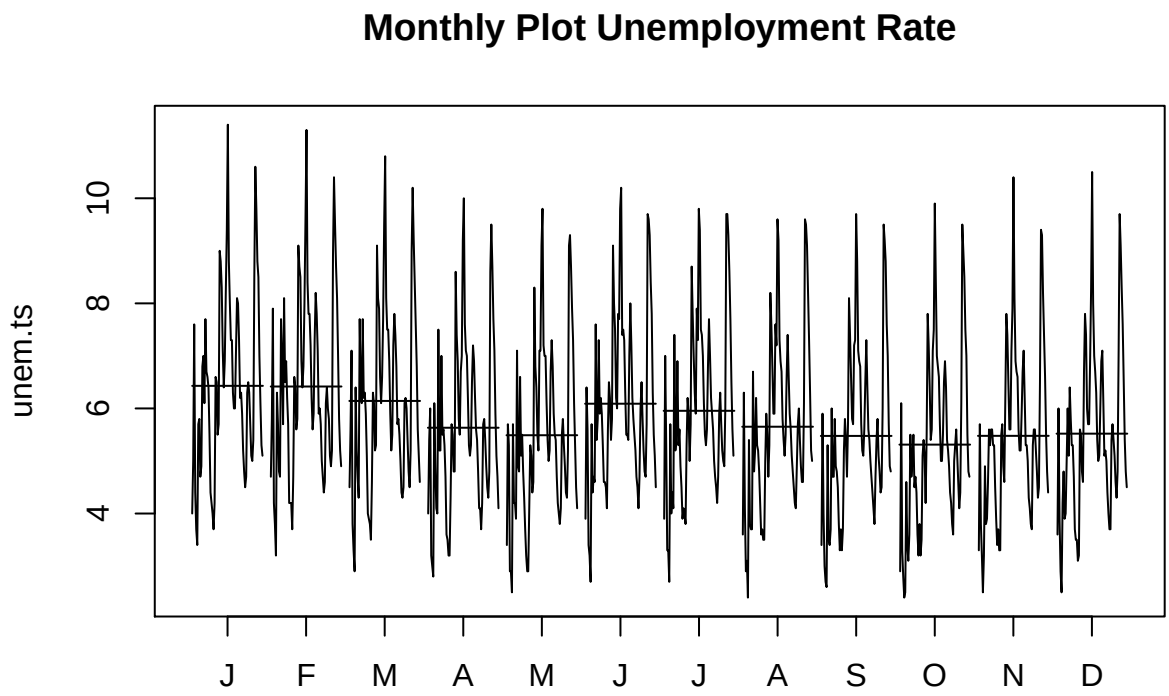


The trend component observed in the decomposed series concurs with our intuitions from the smoothed series. With the seasonal component taken out, the random component shows a spike in mid-1980s and several more in 2000s, both of which may correspond to respective oil price fluctuations. In general, the random component series shows higher variance before mid-1980s than after, which we have also observed in the unemployment series. The decomposed components are clearly not stationary, which again suggest that OLS is not appropriate.

Establish Stationarity

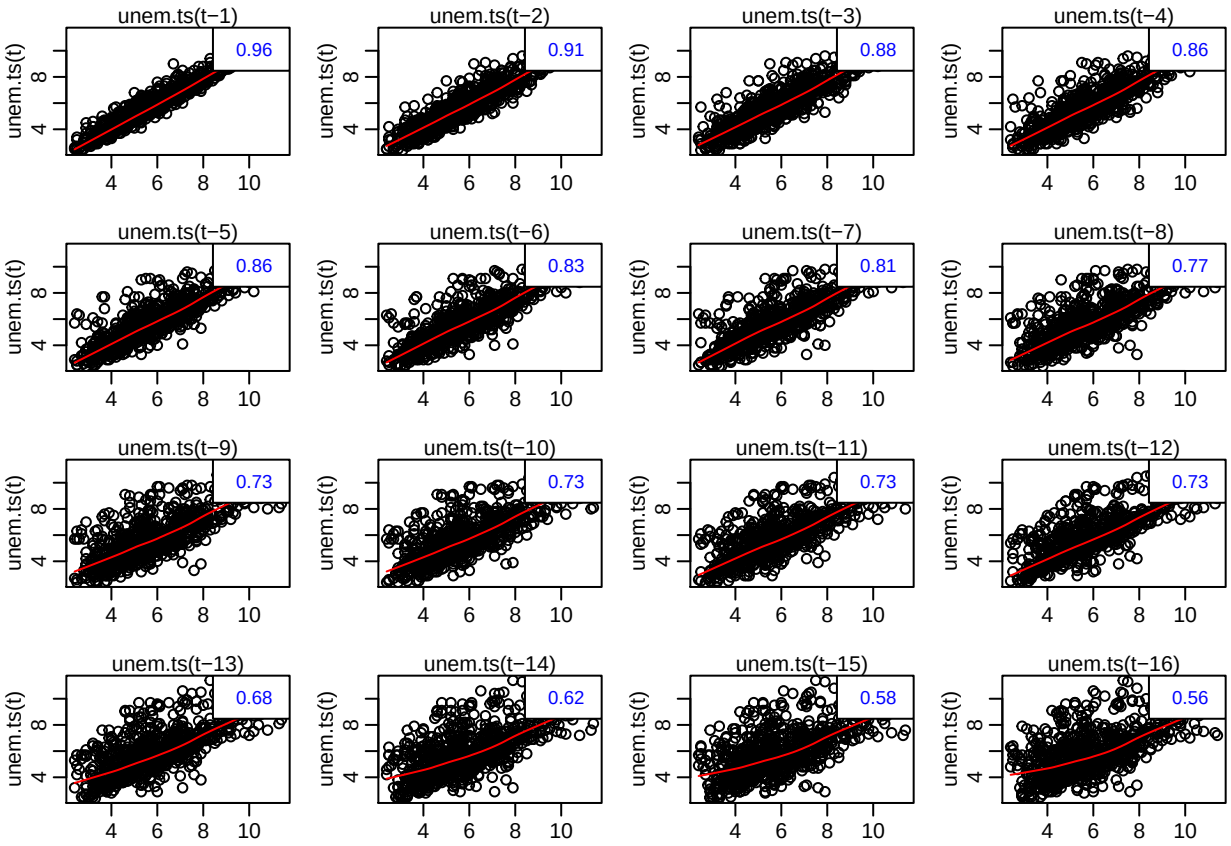
Seasonality and Unit Root Investigation - Unemployment Rate

```
monthplot(unem.ts)
title("Monthly Plot Unemployment Rate")
```



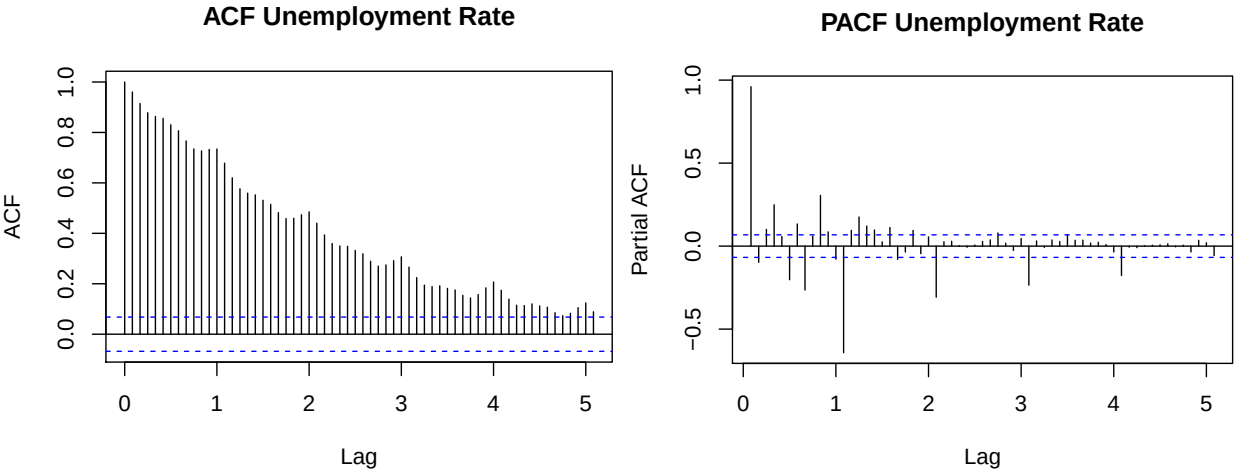
The monthly plot shows some seasonal pattern at the mean, but the range of unemployment variation for each month also overlaps a lot with other months. So the seasonal pattern exists but is not very strong. We see that the unemployment rate tends to be a little higher in the beginning and middle of each year followed by mild gradual decrease.

```
astsa::lag1.plot(unem.ts, 16)
```



The above scatterplot shows that the series is most correlated with its first lag, then correlations grow weaker and the point clouds becomes more scattered with larger lags, without any sign of picking up at lag 12. This plot itself doesn't suggest seasonal effects.

```
acf(unem.ts, lag.max = 61, main = "")
title("ACF Unemployment Rate")
pacf(unem.ts, lag.max = 61, main = "")
title("PACF Unemployment Rate")
```



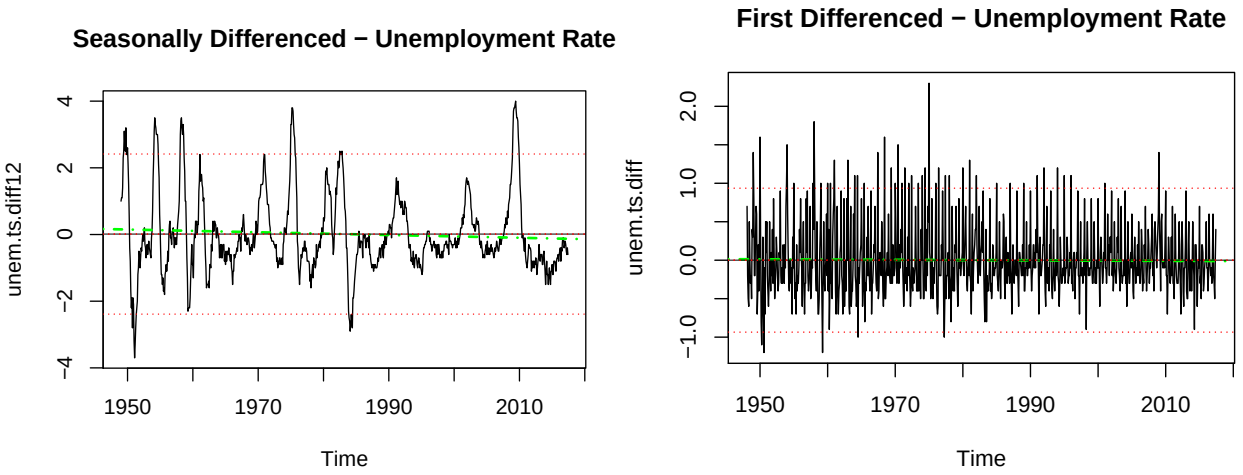
The ACF plot shows slow decay from lag 0 coupled with a sharp drop on the PACF after lag 1. This is a sign of an AR(1) process. We see minor local maxima on the acf at lag 12, 24, 36 and 48 which indicate seasonal effects. The significant, negative spikes on the PACF plot which tail off at lag 13, 25, 37, and 49 suggest seasonal AR or MA components. Some smaller significant pacf values within the first 12 lags suggest either

some AR processes or an MA process in that range.

To help us further in establishing stationarity, we compare the seasonally differenced and first differenced series and compare their behaviors using time and autocorrelation plots.

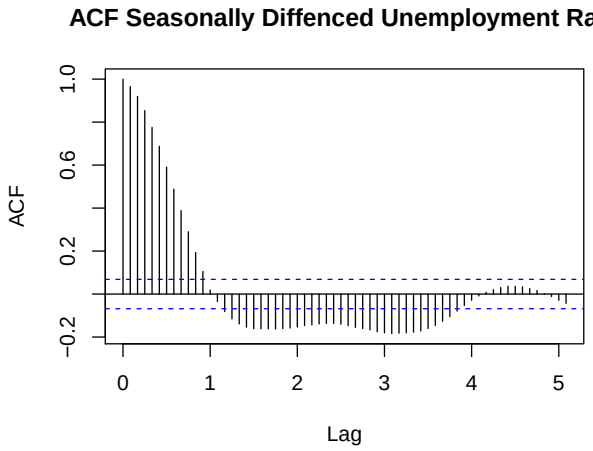
```
unem.ts.diff12 = diff(unem.ts, lag = 12)
unem.ts.diff = diff(unem.ts, lag = 1)
unem.ts.diff.diff12 = diff(unem.ts.diff, lag = 12)
```

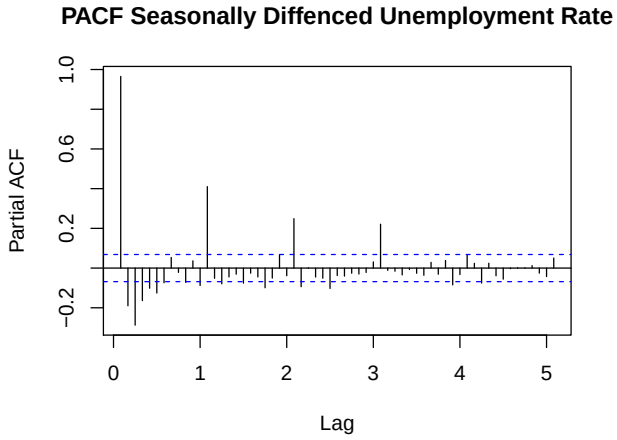
```
ts.plot(unem.ts.diff12)
abline(lm(unem.ts.diff12 ~ time(unem.ts.diff12)), col = "green",
    lty = "dotdash", lwd = 2)
abline(h = mean(unem.ts.diff12), lwd = 0.5)
abline(h = c(mean(unem.ts.diff12), mean(unem.ts.diff12) +
    2 * sd(unem.ts.diff12)), col = "red", lwd = 1, lty = "dotted")
abline(h = c(mean(unem.ts.diff12), mean(unem.ts.diff12) -
    2 * sd(unem.ts.diff12)), col = "red", lwd = 1, lty = "dotted")
title("Seasonally Differenced - Unemployment Rate")

ts.plot(unem.ts.diff)
abline(lm(unem.ts.diff ~ time(unem.ts.diff)), col = "green",
    lty = "dotdash", lwd = 2)
abline(h = mean(unem.ts.diff), lwd = 0.5)
abline(h = c(mean(unem.ts.diff), mean(unem.ts.diff) + 2 *
    sd(unem.ts.diff)), col = "red", lwd = 1, lty = "dotted")
abline(h = c(mean(unem.ts.diff), mean(unem.ts.diff) - 2 *
    sd(unem.ts.diff)), col = "red", lwd = 1, lty = "dotted")
title("First Differenced - Unemployment Rate")
```



The seasonally differenced series retained most of the random walk like behavior in the raw series and appear more persistent before mid 1980s. The first differenced series eliminated most random walk and persistent behaviors but variance is generally larger before 1980. There is still a slight downward trend (green regression line) with the seasonally differenced series. (The red line refers to 2 standard deviations.)

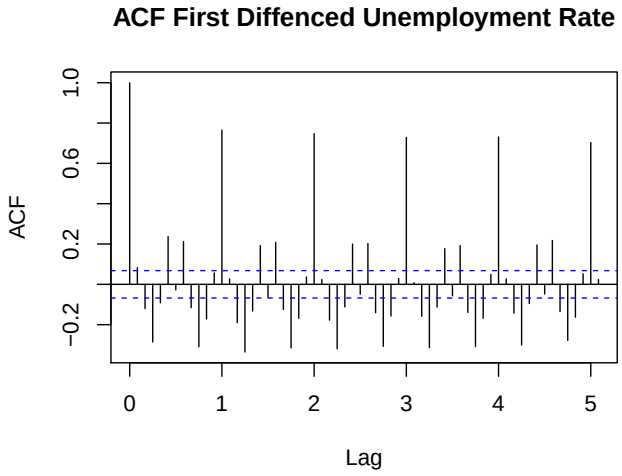
```
acf(unem.ts.diff12, lag.max = 61, main = "")
title("ACF Seasonally Diffenced Unemployment Rate")
pacf(unem.ts.diff12, lag.max = 61, main = "")
title("PACF Seasonally Diffenced Unemployment Rate")
```

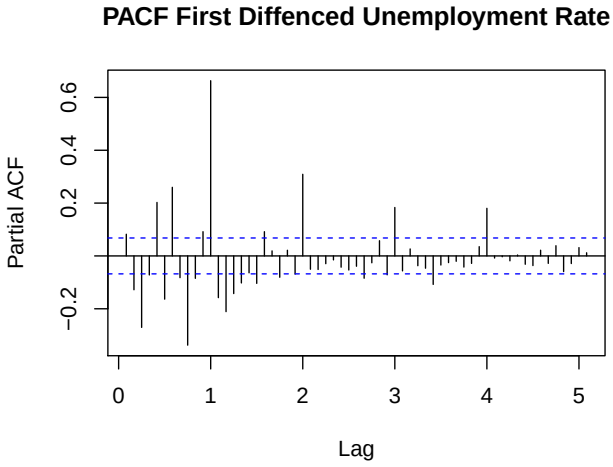


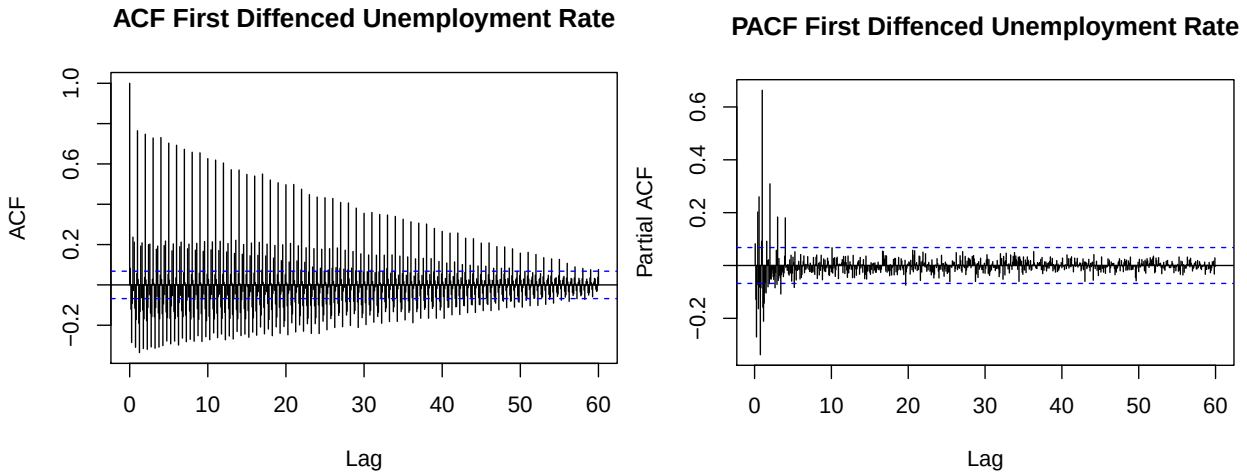


In the seasonally differenced series, we still see a significant pacf at lag 1 followed by a sharp drop and gradual decay in the acf. Notice the significant pacfs in lags 13, 25, and 37 as well. This suggests an AR(1) process in combination with seasonal AR(3), or in combination with some seasonal ARMA processes. Notice a couple significant pacfs between lag 2 to 4; they can be AR(2-4) components but it is hard to tell at this stage. We may entertain the model SARIMA(4,0,q)(3,1,Q) from these plots.

```
acf(unem.ts.diff, lag.max = 61, main = "")
title("ACF First Diffenced Unemployment Rate")
pacf(unem.ts.diff, lag.max = 61, main = "")
title("PACF First Diffenced Unemployment Rate")

# Additional Lags
acf(unem.ts.diff, lag.max = 720, main = "")
title("ACF First Diffenced Unemployment Rate")
pacf(unem.ts.diff, lag.max = 720, main = "")
title("PACF First Diffenced Unemployment Rate")
```

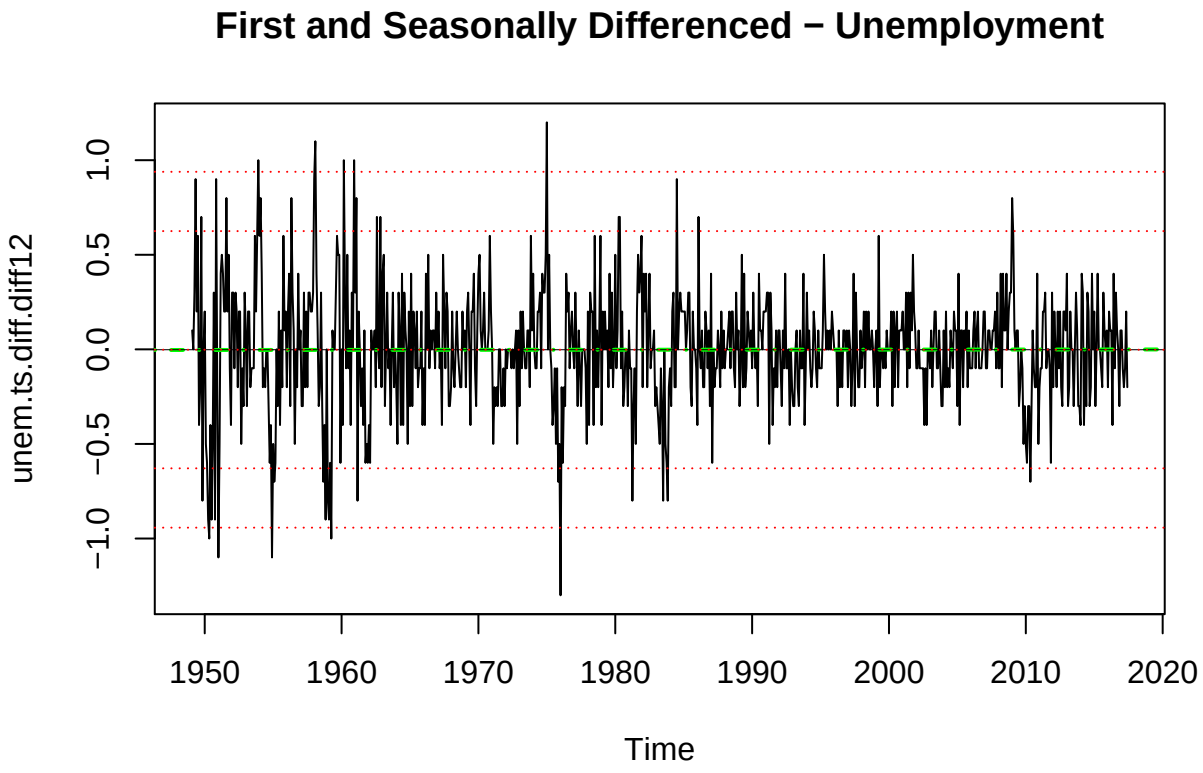






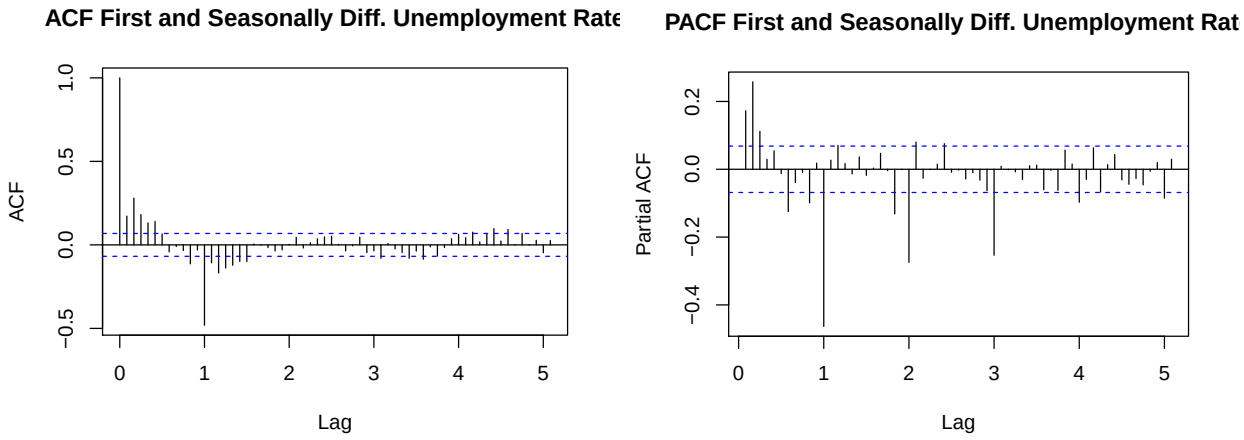
In the first differenced series, the plots show periodic behavior. The extended plot with additional lags shows that the acf tails off until lag 720, which suggest either a seasonal long memory or seasonal integrated process (both SAR(1)) in combination with some sesonal AR(2-4) components or in combination with a seasonal MA component. The pacf shows some significance between lag 2-9 without a little echo in the acf lag 13-16, which can come from some MA processes, but it's hard to say at this stage. We will assume there are some ARMA processes present. We may entertain the model as a SARIMA(9,1,q)(4,0,Q) using these plots.

```
ts.plot(unem.ts.diff.diff12)
abline(lm(unem.ts.diff.diff12 ~ time(unem.ts.diff.diff12)),
    col = "green", lty = "dotdash", lwd = 2)
abline(h = mean(unem.ts.diff.diff12), lwd = 0.5)
abline(h = c(mean(unem.ts.diff.diff12), mean(unem.ts.diff.diff12) +
    c(-3, -2, 2, 3) * sd(unem.ts.diff.diff12)), col = "red",
    lwd = 1, lty = "dotted")
title("First and Seasonally Differenced - Unemployment")
```



The first and seasonally differenced series seem more volatile before mid 1960s. Nevertheless, the fluctuation intervals and magnitude of spikes are similar across the whote time series. This series appear stationary at the mean and most observations stay within 2 deviations.

```
acf(unem.ts.diff.diff12, lag.max = 61, main = "")
title("ACF First and Seasonally Diff. Unemployment Rate")
pacf(unem.ts.diff.diff12, lag.max = 61, main = "")
title("PACF First and Seasonally Diff. Unemployment Rate")
```



In the first and seasonally differenced series, the autocorrelation plots don't show an AR(1) component anymore. We still see a significant acf at lag 12 and significant decaying pacf at lag 12, 24, 36 and 48, which suggests a seasonal MA(1) process. At lag 2-4, the pacf are still significant with some echos in acf which suggests some AR(4) components. We will assume some ARMA processes present. We may entertain the model as a SARIMA(4,1,0)(0,1,1) using these plots.

Performing both differencing procedures may over-difference the series. To determine how much is enough, we perform unit root tests below to check for stationarity. Augmented Dicky Fuller Test and Phillips Perron Tests are performed, with the following test hypotheses:

- Ho : The series has a unit root
- Ha : The series is stationary

In the ADF test, our null hypothesis assume that the process is a random walk with drift and some AR(p) components,$x_t = \beta_0 + \phi x_{t-1} + \sum_{j=1}^{p-1} \psi_j x_{t-j} + w_t$, where β_0 represents the drift and $\phi = 1$. We are essentially testing the null hypothesis $\gamma = 0$ in the differenced series $\nabla x_t = \gamma x_{t-1} + \sum_{j=1}^{p-1} \psi_j \nabla x_{t-j} + w_t$ since $\gamma = \phi - 1$. Under the alternative hypothesis $\gamma < 0$. On the other hand, the Philips Perron test, our null hypothesis assume a model $x_t = \beta_0 + \rho_1 x_{t-1} + u_t$, where non-parametric correction is applied on ρ to correct for serial correlation in u_t already. We are essential testing of $\rho = 1$ in the null hypothesis.

```
tseries::adf.test(unem.ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  unem.ts
## Dickey-Fuller = -2.5911, Lag order = 9, p-value = 0.3281
## alternative hypothesis: stationary
```

```
tseries::pp.test(unem.ts)
```

```
##
##  Phillips-Perron Unit Root Test
##
```

```
## data:  unem.ts
## Dickey-Fuller Z(alpha) = -28.648, Truncation lag parameter = 6,
## p-value = 0.01093
## alternative hypothesis: stationary
```

```
tseries::adf.test(unem.ts.diff)
```

```
## Warning in tseries::adf.test(unem.ts.diff): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  unem.ts.diff
## Dickey-Fuller = -13.13, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(unem.ts.diff)
```

```
## Warning in tseries::pp.test(unem.ts.diff): p-value smaller than printed p-
## value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  unem.ts.diff
## Dickey-Fuller Z(alpha) = -602.1, Truncation lag parameter = 6,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(unem.ts.diff12)
```

```
## Warning in tseries::adf.test(unem.ts.diff12): p-value smaller than printed
## p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  unem.ts.diff12
## Dickey-Fuller = -8.1444, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(unem.ts.diff12)
```

```
## Warning in tseries::pp.test(unem.ts.diff12): p-value smaller than printed
## p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  unem.ts.diff12
## Dickey-Fuller Z(alpha) = -60.968, Truncation lag parameter = 6,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(unem.ts.diff.diff12)
```

```
## Warning in tseries::adf.test(unem.ts.diff.diff12): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  unem.ts.diff.diff12
## Dickey-Fuller = -9.3802, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(unem.ts.diff.diff12)
```

```
## Warning in tseries::pp.test(unem.ts.diff.diff12): p-value smaller than
## printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  unem.ts.diff.diff12
## Dickey-Fuller Z(alpha) = -894.88, Truncation lag parameter = 6,
## p-value = 0.01
## alternative hypothesis: stationary
```

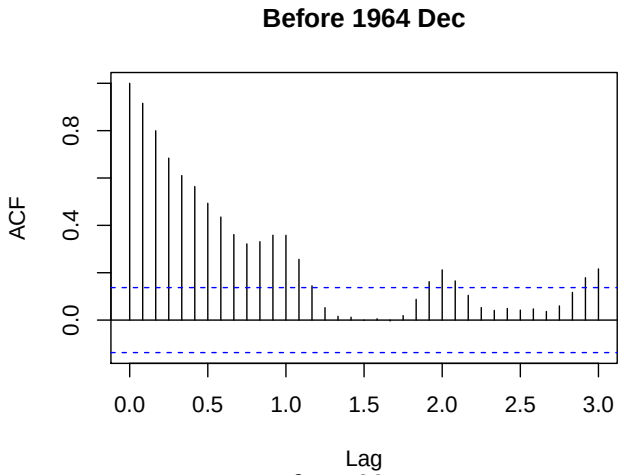
As expected with the raw series, the ADF test failed to reject the null hypothesis that the series contains a unit root. The PP test results contradicts, but we acknowledge that the test mechanism is different and that some early literature (Davidson and Mackinnon 2004) showed that ADF performs better in finite sample than PP test. All the tests rejected the null hypotheses that either the first differenced or seasonally differenced or first and seasonally differenced series contains a unit root and supports stationarity. The unit root tests by themselves don't suggest both first and seasonal differencing.

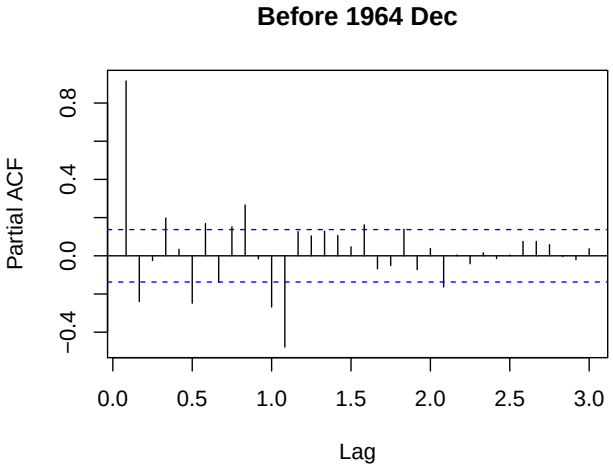
Unemployment Series Stationarity Conclusion

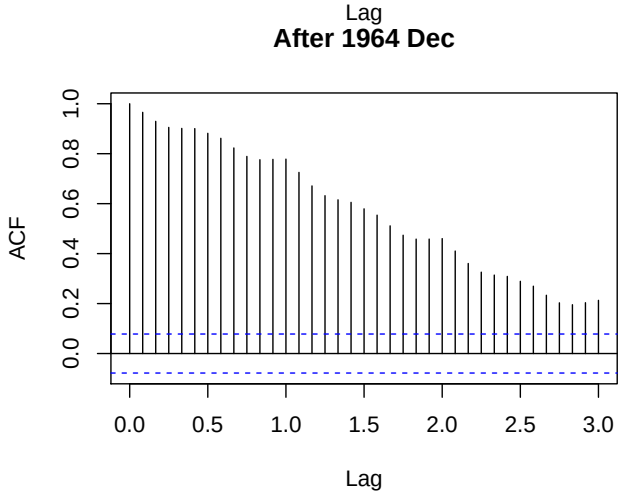
Although some plots show more volatile behavior of the series before mid-1960s, we decide not to truncate the series because the random walk behavior, seasonal pattern, spike magnitude, and slight upward trend seem consistent through the entire series. We believe the generating process underlying unemployment rate is similar enough from 1948 to 2017 that keeping all observations will provide more precise estimates. Referring to the correlation plots below, the acf and pacf plots of the series before and after mid-1960s are fairly similar. Both split series exhibit strong AR(1) behavior and some seasonal patterns. (Note that the shorter decay in the split series before 1964 can be attributed to a substantially smaller sample size.)

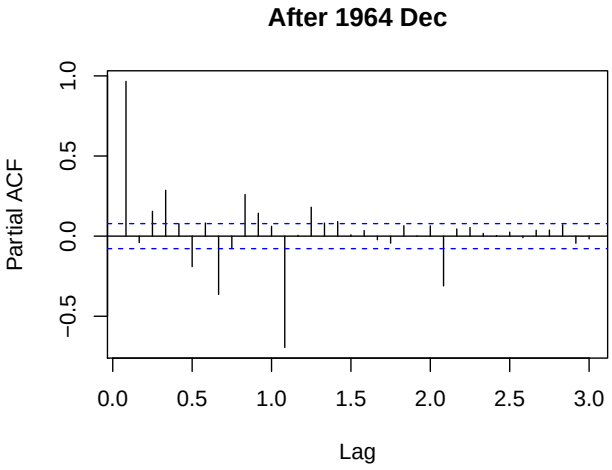
Regarding first differencing, all of our plots and tests suggests strong random walk behavior with the raw series, so it is recommended. Regarding additional seasonal differencing, the results were inconclusive. The monthly, acf and pacf plots suggest noticeable seasonal pattern, while the scatterplot matrix doesn't. The unit root test suggested that either first or seasonal differencing can help us to establish a stationary series. At this stage, we remain open to a first differenced, or first differenced in additional to seasonally differenced model. We defer to the modeling and forecasting results to determine the best candidate.

```
unem.ts.til1964 = window(x = unem.ts, end = c(1964, 12))
unem.ts.from1964 = window(x = unem.ts, start = c(1965, 1))

acf(unem.ts.til1964, main = "", lag.max = 36)
title("Before 1964 Dec")
pacf(unem.ts.til1964, main = "", lag.max = 36)
title("Before 1964 Dec")
acf(unem.ts.from1964, main = "", lag.max = 36)
title("After 1964 Dec")
pacf(unem.ts.from1964, main = "", lag.max = 36)
title("After 1964 Dec")
```





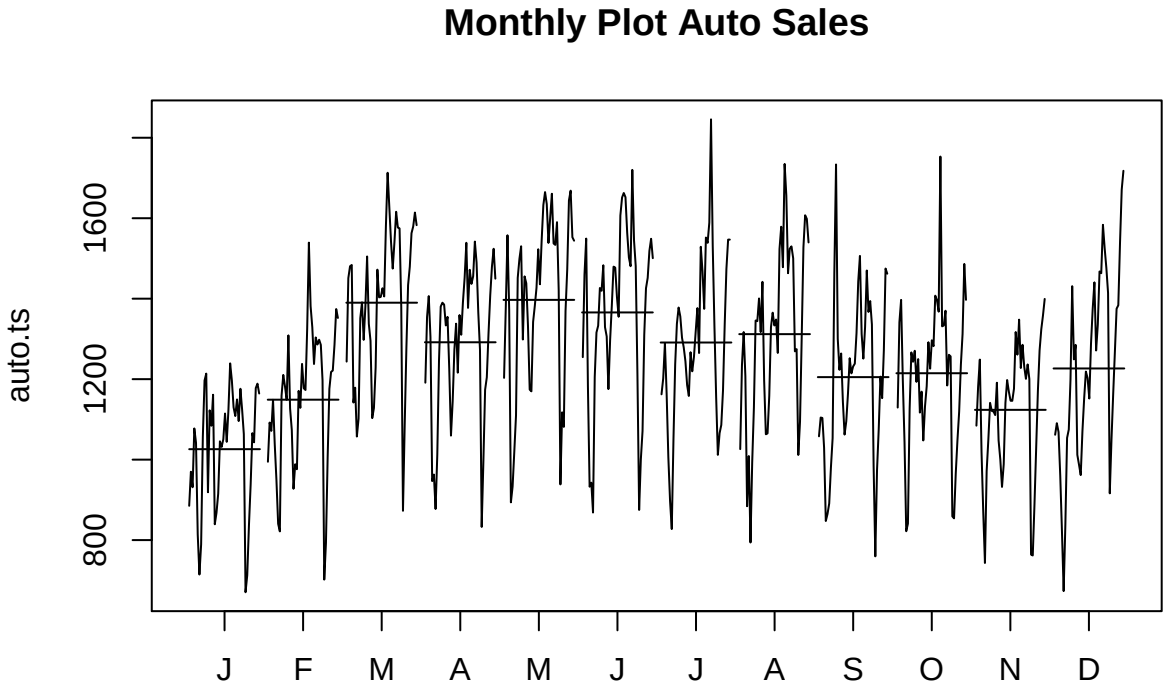




Seasonality and Unit Root Investigation – Auto Sales

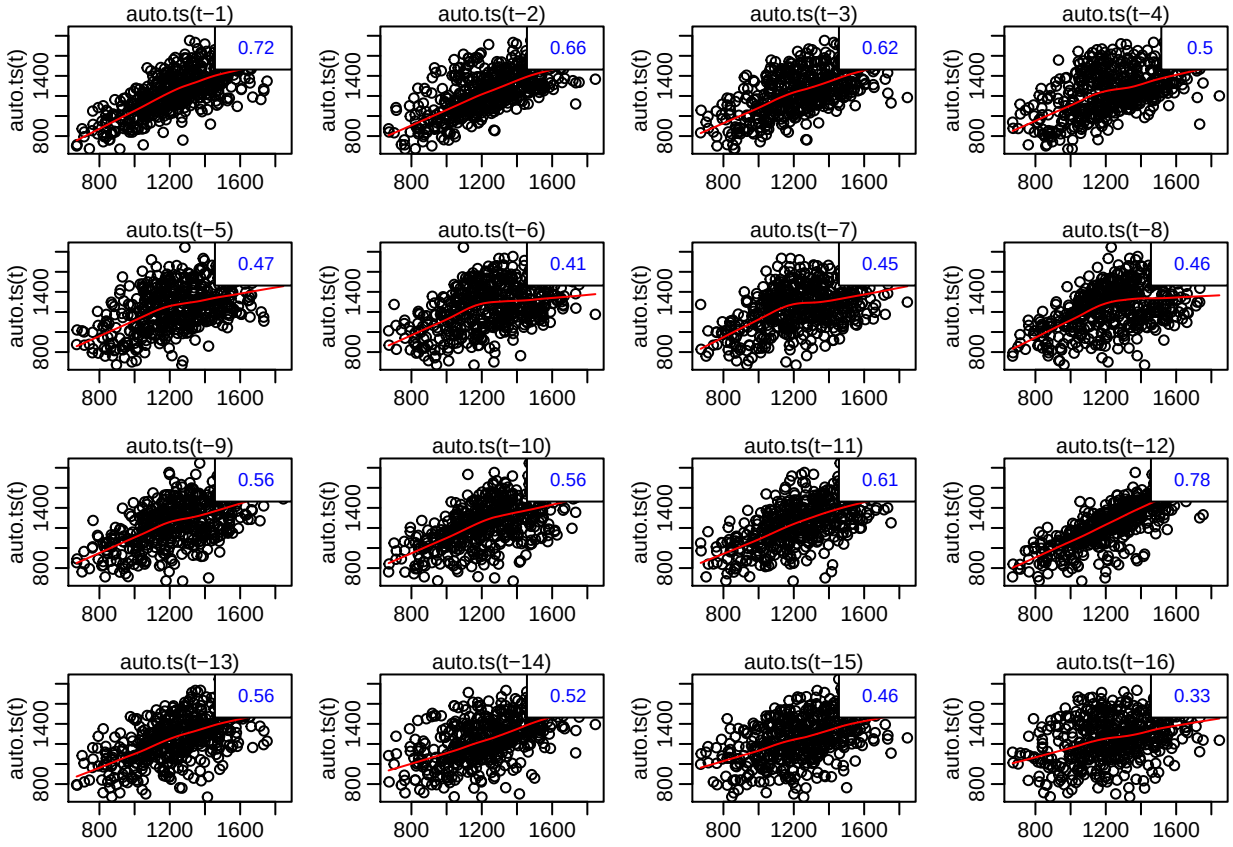
Next we will conduct a similar analysis of the auto sales dataset.

```
monthplot(auto.ts)
title("Monthly Plot Auto Sales")
```



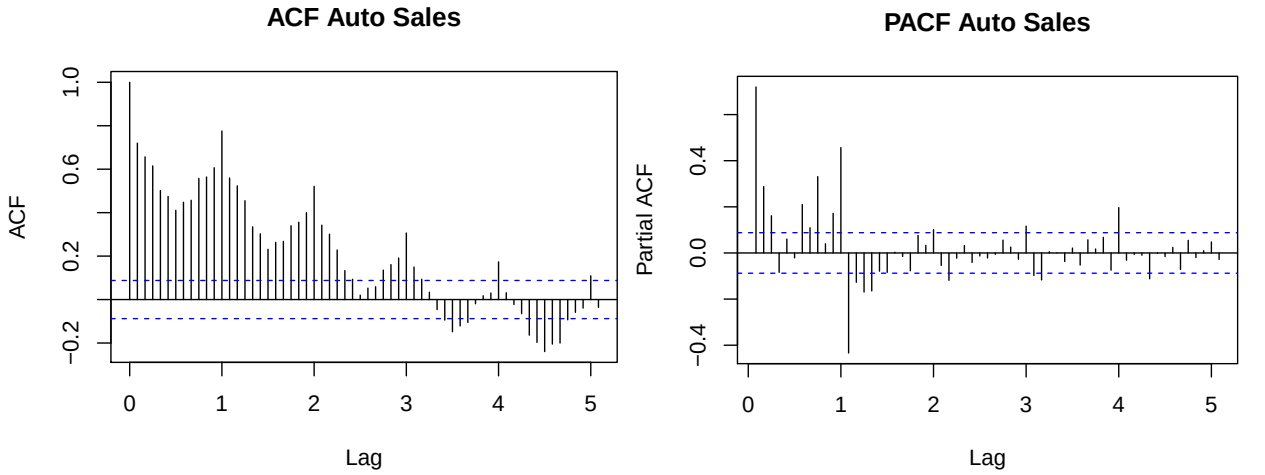
Compared to that of unemployment rate, the monthly plot of auto sales shows much more discrete values in the mean for each month. The seasonal pattern is much stronger. We see that auto sales tend to be noticeably lower in the first two months, pick up sharply in March, then step up and down for the rest of the year. Fall sales are generally lower than summer sales.

```
astsa::lag1.plot(auto.ts, 16)
```



The scatterplot matrix of auto sales with its own lags displays clear seasonal dependence. Autocorrelation is highest with lag 12, which is the same month last year, even more than when compared to the previous month. The opposite patterns were observed in the unemployment rates series, when autocorrelation was highest with lag 1 and then gradually declines for higher lags.

```
acf(auto.ts, lag.max = 61, main = "")
title("ACF Auto Sales")
pacf(auto.ts, lag.max = 61, main = "")
title("PACF Auto Sales")
```



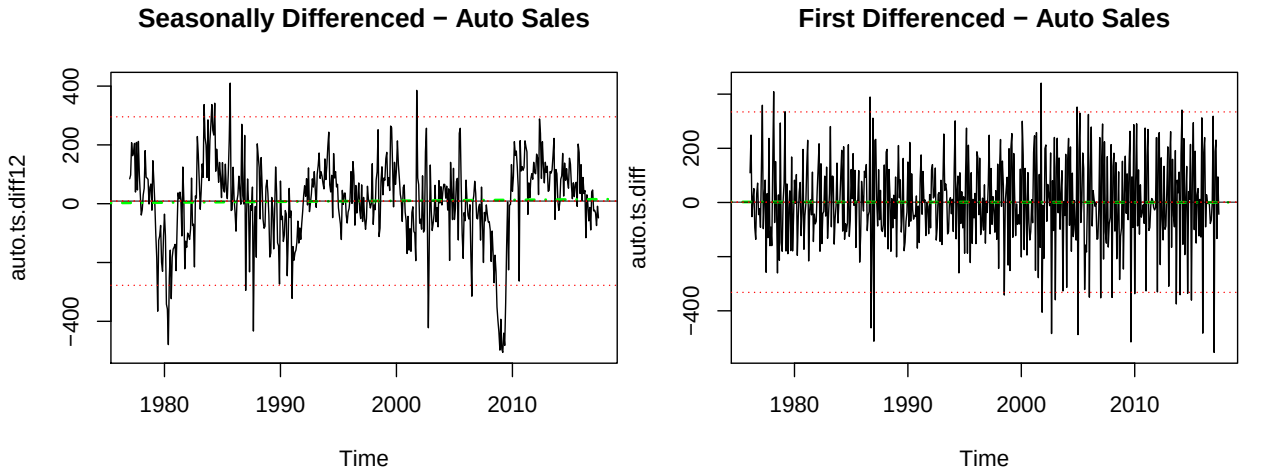
Similar to the unemployment series, the acf shows slow decay from lag 0, a sharp drop of pacf after lag 1 and some local acf maximums at lag 12, 24, 36 and 48. Unlike the unemployment series, the acf decay ends earlier at lag 36. The local maxima have much stronger profiles. Some significant pacf values occur before lag 12 which suggest either some AR or MA components (effects on acf plot are not discernable) in that range.

Based on a strong seasonal pattern in the above three plots, seasonal differencing would be useful in achieving stationarity. To help us determine, we again compare the seasonally differenced and first differenced series and compare their behaviors using time and autocorrelation plots.

```
auto.ts.diff12 = diff(auto.ts, lag = 12)
auto.ts.diff = diff(auto.ts, lag = 1)
auto.ts.diff.diff12 = diff(auto.ts.diff, lag = 12)
```

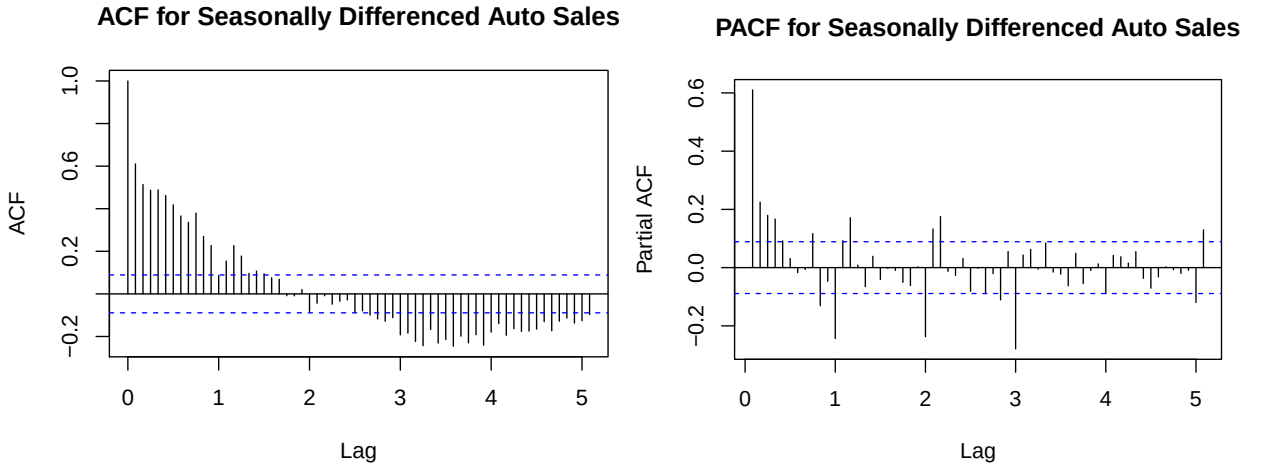
```
ts.plot(auto.ts.diff12)
abline(lm(auto.ts.diff12 ~ time(auto.ts.diff12)), col = "green",
    lty = "dotdash", lwd = 2)
abline(h = mean(auto.ts.diff12), lwd = 0.5)
abline(h = c(mean(auto.ts.diff12), mean(auto.ts.diff12) +
    2 * sd(auto.ts.diff12)), col = "red", lwd = 1, lty = "dotted")
abline(h = c(mean(auto.ts.diff12), mean(auto.ts.diff12) -
    2 * sd(auto.ts.diff12)), col = "red", lwd = 1, lty = "dotted")
title("Seasonally Differenced - Auto Sales")

ts.plot(auto.ts.diff)
abline(lm(auto.ts.diff ~ time(auto.ts.diff)), col = "green",
    lty = "dotdash", lwd = 2)
abline(h = mean(auto.ts.diff), lwd = 0.5)
abline(h = c(mean(auto.ts.diff), mean(auto.ts.diff) + 2 *
    sd(auto.ts.diff)), col = "red", lwd = 1, lty = "dotted")
abline(h = c(mean(auto.ts.diff), mean(auto.ts.diff) - 2 *
    sd(auto.ts.diff)), col = "red", lwd = 1, lty = "dotted")
```

```
title("First Differenced - Auto Sales")
```



The seasonally differenced series noticeably removed the strong seasonal patterns and some degree of persistency from the raw series, but the random walk behavior of raw series before early 1990s and after mid 2000s is retained. The first differenced series removed most persistencies from the raw series but appear clustered regularly at seasonally intervals. Both series removed the upward trend in the raw series effectively.

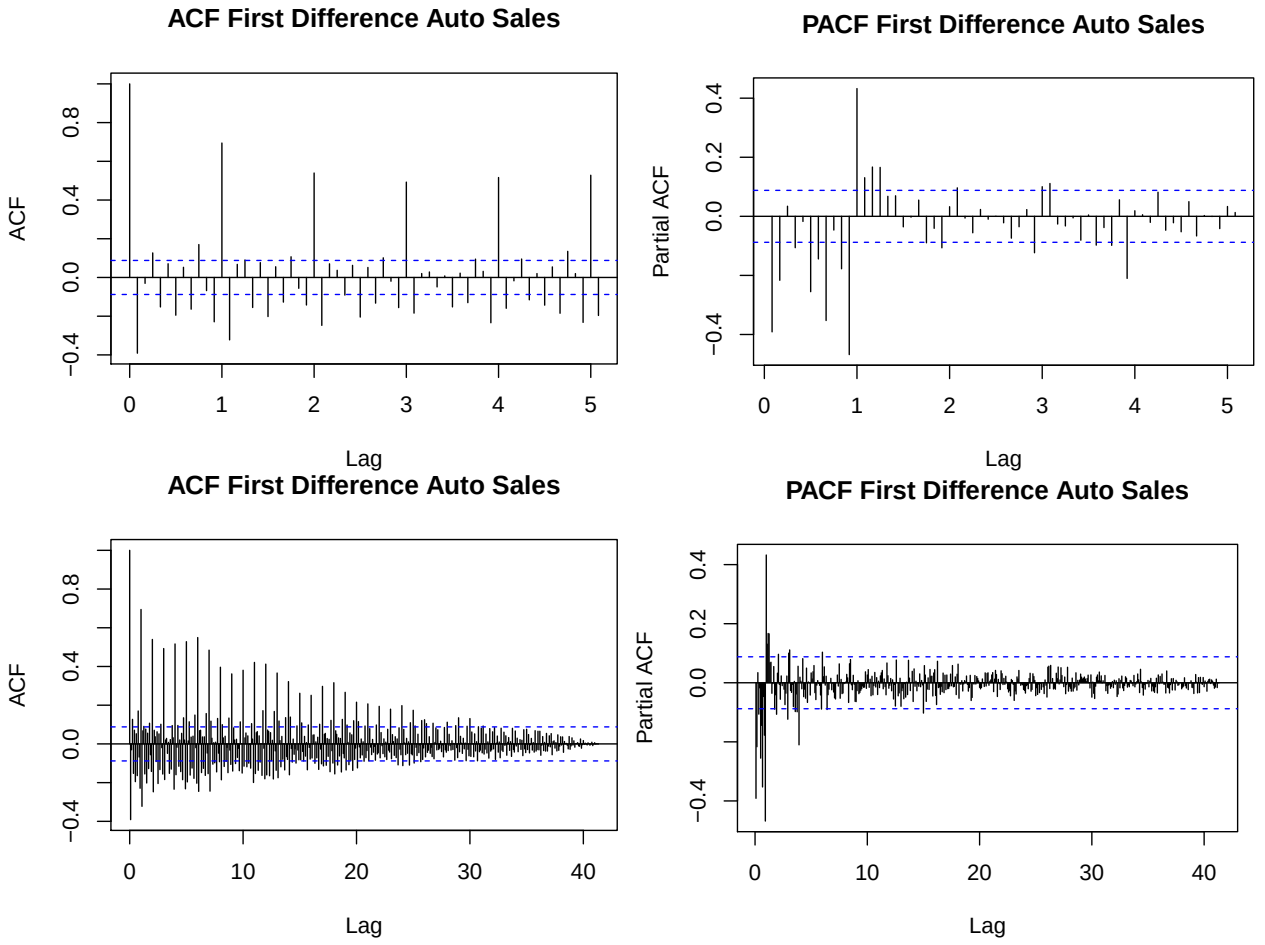
```
acf(auto.ts.diff12, lag.max = 61, main = "")
title("ACF for Seasonally Differenced Auto Sales")
pacf(auto.ts.diff12, lag.max = 61, main = "")
title("PACF for Seasonally Differenced Auto Sales")
```



Similar to unmployment rate, in the seasonally differenced series of auto sales, we still see a significant pacf at lag 1 followed by a sharp drop and gradual decay in the acf. Notice the significant pacfs in lag 12, 24 and 36. This suggests an AR(1) process in combination with seasonal AR(3) or some seasonal MA component. Also notice a few significant pacfs at lags 2-4. We may entertain the model as a SARIMA(4,0,q)(3,1,Q) using these plots. It doesn't seem like an additional first differencing is necessary.

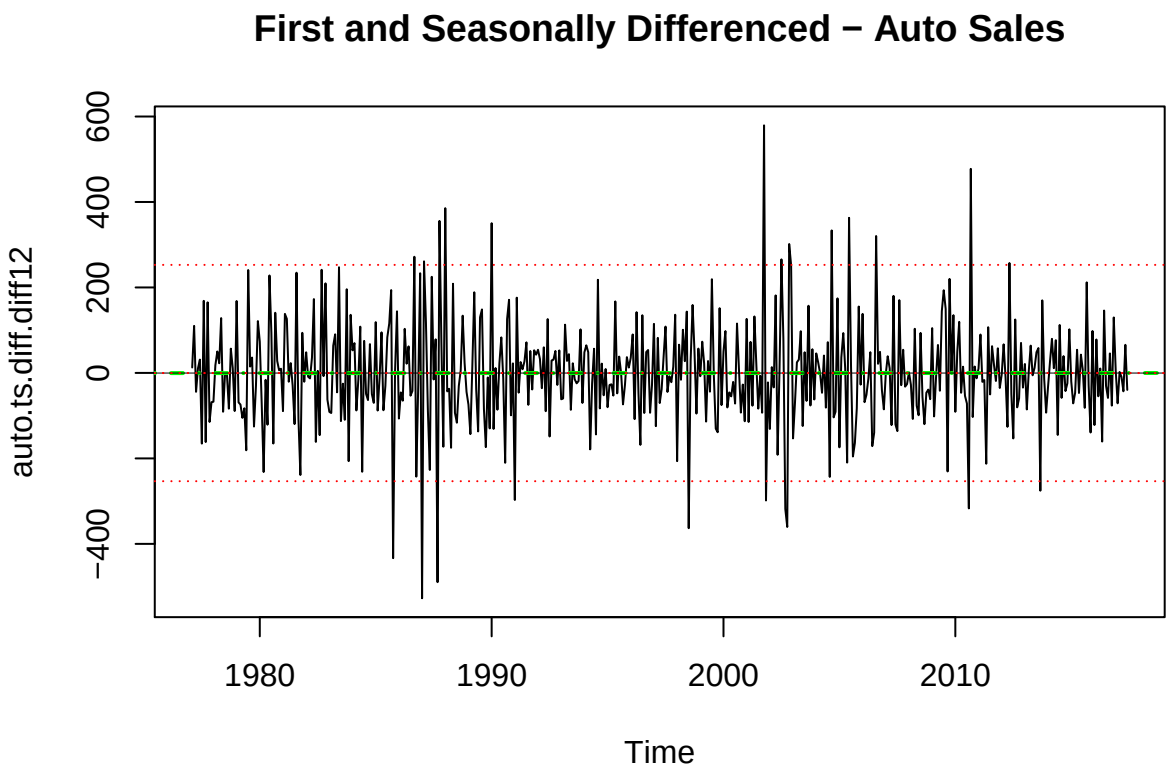
```
acf(auto.ts.diff, lag.max = 61, main = "")
title("ACF First Difference Auto Sales")
pacf(auto.ts.diff, lag.max = 61, main = "")
title("PACF First Difference Auto Sales")

# Additional Lags
acf(auto.ts.diff, lag.max = 720, main = "")
```

```
title("ACF First Difference Auto Sales")
pacf(auto.ts.diff, lag.max = 720, main = "")
title("PACF First Difference Auto Sales")
```



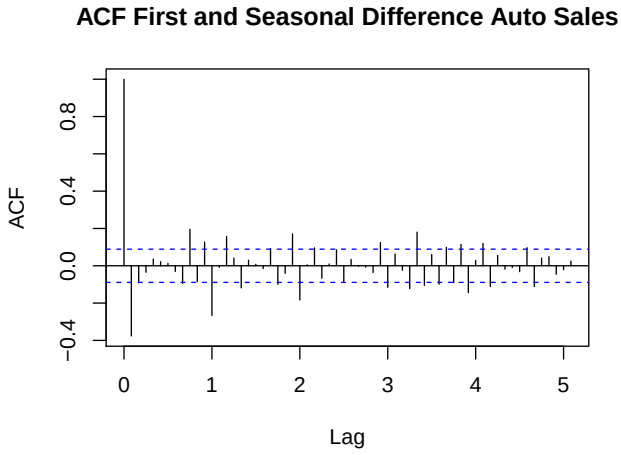
Similar to unmployment rate, in the first differenced series of auto sales, the acf tails off very slowly until lag 360. Unlike that of the unemployment series, the seasonal ripples on the acf appear much stronger. This plots suggest a strong seasonal AR(1) component and some AR(p) components before lag 12.

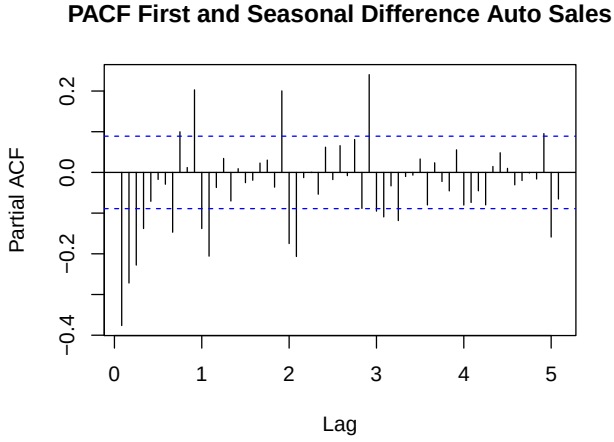
```
ts.plot(auto.ts.diff.diff12)
abline(lm(auto.ts.diff.diff12 ~ time(auto.ts.diff.diff12)),
    col = "green", lty = "dotdash", lwd = 2)
abline(h = mean(auto.ts.diff.diff12), lwd = 0.5)
abline(h = c(mean(auto.ts.diff.diff12), mean(auto.ts.diff.diff12) +
    2 * sd(auto.ts.diff.diff12)), col = "red", lwd = 1,
    lty = "dotted")
abline(h = c(mean(auto.ts.diff.diff12), mean(auto.ts.diff.diff12) -
    2 * sd(auto.ts.diff.diff12)), col = "red", lwd = 1,
    lty = "dotted")
title("First and Seasonally Differenced - Auto Sales")
```



Despite several spikes, the first and seasonally differenced series seems roughly stationary at the mean and variance. Most observations stay within two standard deviations.

```
acf(auto.ts.diff.diff12, lag.max = 61, main = "")
title("ACF First and Seasonal Difference Auto Sales")
pacf(auto.ts.diff.diff12, lag.max = 61, main = "")
title("PACF First and Seasonal Difference Auto Sales")
```





In the first and seasonally differenced series, the autocorrelation plots don't show an AR(1) component anymore, we still see significant acf and pacf around lag 12, 24, 36, which indicate some seasonal ARMA components. We also see a significant acf at lag 1 echoed by some significant pacf which indicate an MA(1) component. We may entertain the model as a SARIMA(1,1,0)(P,1,Q) using these plots. Notice that the occurence of both MA and SMA components could come from over-differencing the raw series. But if our goal is to produce better forecasts rather than estimating a random component detrended from the raw series, first in additional to seasonal differencing is still acceptable.

We perform unit root tests below to check for stationarities of the raw, first differenced and seasonally

differenced series for auto sales. Augmented Dickey Fuller Test and Phillips Perron Tests are performed, with the following test hypotheses:

- Ho : The series has a unit root
- Ha : The series is stationary

```
tseries::adf.test(auto.ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  auto.ts
## Dickey-Fuller = -3.5662, Lag order = 7, p-value = 0.03595
## alternative hypothesis: stationary
```

```
tseries::pp.test(auto.ts)
```

```
## Warning in tseries::pp.test(auto.ts): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  auto.ts
## Dickey-Fuller Z(alpha) = -157.6, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(auto.ts.diff)
```

```
## Warning in tseries::adf.test(auto.ts.diff): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  auto.ts.diff
## Dickey-Fuller = -16.651, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(auto.ts.diff)
```

```
## Warning in tseries::pp.test(auto.ts.diff): p-value smaller than printed p-
## value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  auto.ts.diff
## Dickey-Fuller Z(alpha) = -611.4, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(auto.ts.diff12)
```

```
## Warning in tseries::adf.test(auto.ts.diff12): p-value smaller than printed
## p-value
```

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

```
## data:  auto.ts.diff12
## Dickey-Fuller = -4.1696, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(auto.ts.diff12)
```

```
## Warning in tseries::pp.test(auto.ts.diff12): p-value smaller than printed
## p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  auto.ts.diff12
## Dickey-Fuller Z(alpha) = -199.97, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(auto.ts.diff.diff12)
```

```
## Warning in tseries::adf.test(auto.ts.diff.diff12): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  auto.ts.diff.diff12
## Dickey-Fuller = -11.281, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::pp.test(auto.ts.diff.diff12)
```

```
## Warning in tseries::pp.test(auto.ts.diff.diff12): p-value smaller than
## printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  auto.ts.diff.diff12
## Dickey-Fuller Z(alpha) = -541.41, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

All tests rejected the null hypotheses for all four series. Notice that the ADF test p-value for the raw series is closer to the critical cut off of 0.05. There can be a very weak chance that the raw series contains a unit root.

Auto sales series stationarity conclusion:

Regarding seasonal differencing, all of our plots suggests strong seasonal patterns in the raw series, so it is recommended. Regarding first differencing, our raw series and seasonally differenced series both show noticeable AR(1) behavior and the unit test only weakly reject the null hypothesis of stationarity. At this stage, we remain open to a seasonal difference, or first difference in addition to a seasonally differenced model. Again, we defer to the modeling and forecasting results to determine the best candidate.

Examine Bivariate Relationship

Next we examine bivariate relationships between the unemployment rate and auto sales data.

Overview and cross-correlation

```
# Intersect the series
combined.raw = ts.intersect(unem.ts, auto.ts)
unem.intersect.ts = combined.raw[, 1]
auto.intersect.ts = combined.raw[, 2]

head(combined.raw)
```

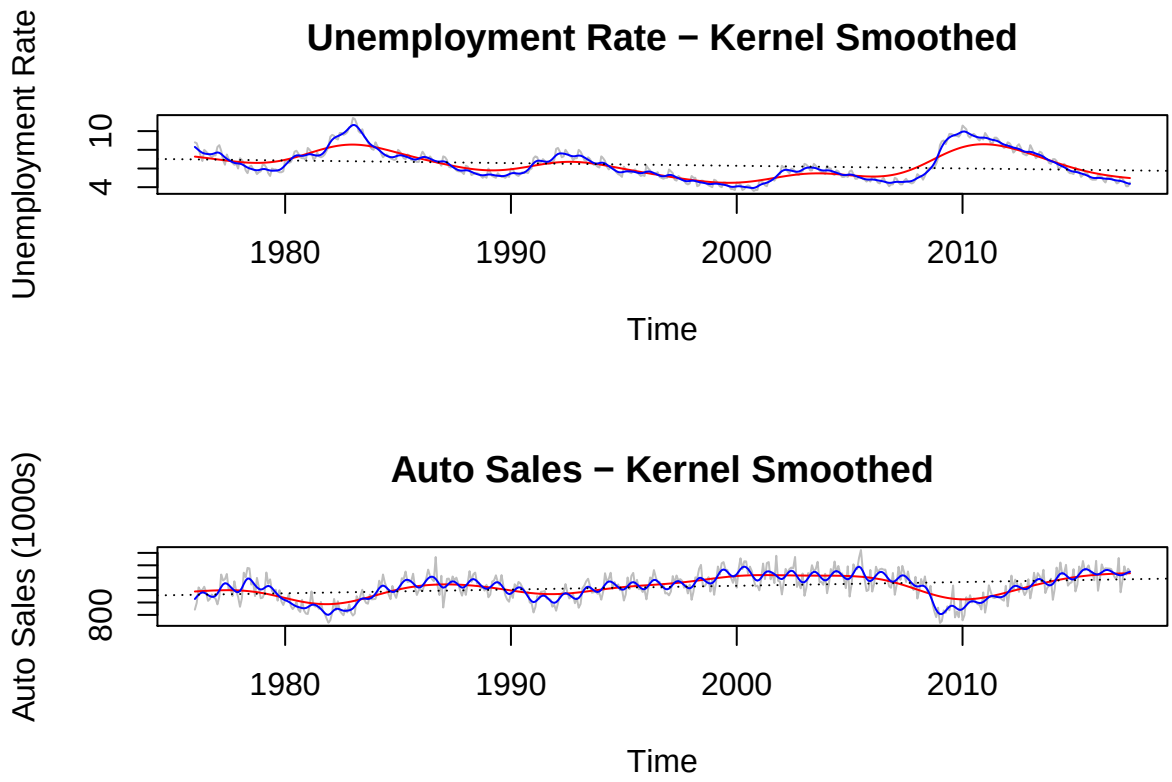
```
##          unem.ts auto.ts
## Jan 1976     8.8   885.2
## Feb 1976     8.7   994.7
## Mar 1976     8.1  1243.6
## Apr 1976     7.4  1191.2
## May 1976     6.8  1203.2
## Jun 1976     8.0  1254.7
```

```
tail(combined.raw)
```

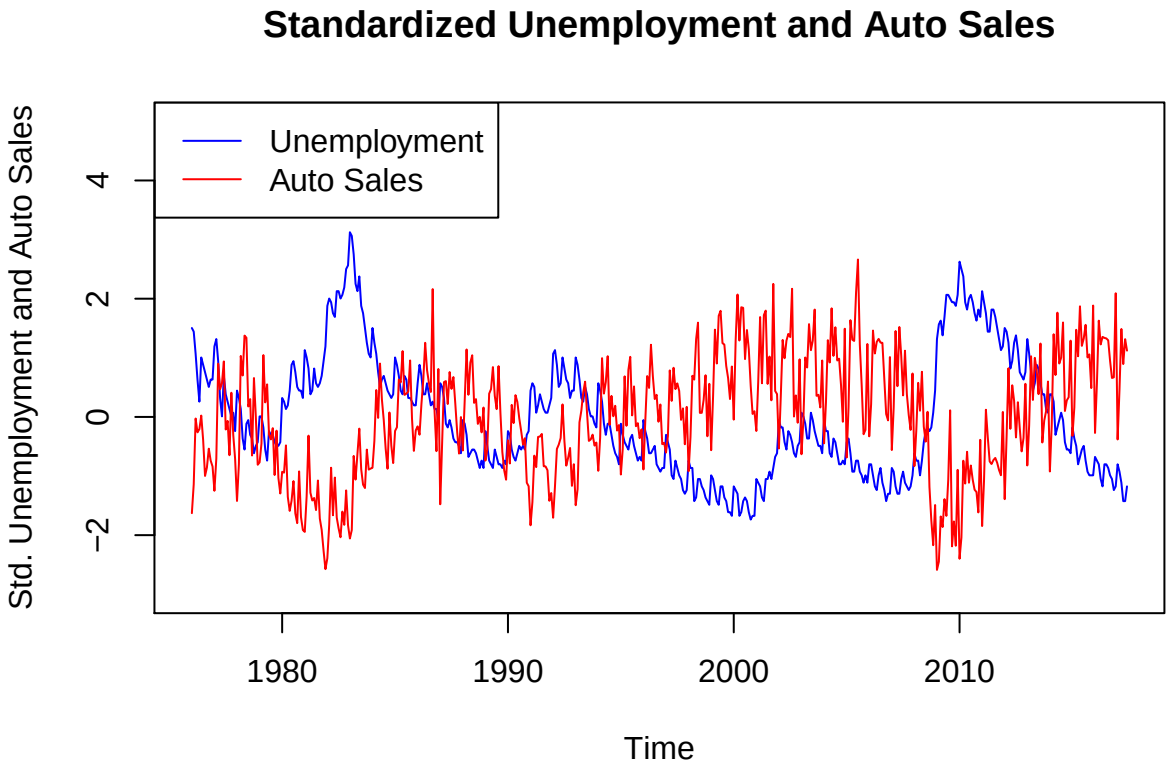
```
##          unem.ts auto.ts
## Jan 2017     5.1  1164.3
## Feb 2017     4.9  1352.1
## Mar 2017     4.6  1582.7
## Apr 2017     4.1  1449.7
## May 2017     4.1  1544.1
## Jun 2017     4.5  1500.6
```

```
par(mfrow = c(2, 1))
# Kernel smoothing
unem.k.smooth.widest = ksmooth(time(unem.intersect.ts),
    unem.intersect.ts, kernel = c("normal"), bandwidth = 10)

unem.k.smooth.wide = ksmooth(time(unem.intersect.ts), unem.intersect.ts,
    kernel = c("normal"), bandwidth = 4)

unem.k.smooth.narrow = ksmooth(time(unem.intersect.ts),
    unem.intersect.ts, kernel = c("normal"), bandwidth = 0.5)

# Make plot
plot(unem.intersect.ts, col = "gray", ylab = "Unemployment Rate",
    main = "Unemployment Rate - Kernel Smoothed")
lines(unem.k.smooth.wide$x, unem.k.smooth.wide$y, col = "red")
lines(unem.k.smooth.narrow$x, unem.k.smooth.narrow$y, col = "blue")
abline(lm(unem.intersect.ts ~ time(unem.intersect.ts)),
    lty = "dotted", col = "black")

# Kernel smoothing
auto.k.smooth.widest = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 10)

auto.k.smooth.wide = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 4)

auto.k.smooth.narrow = ksmooth(time(auto.ts), auto.ts, kernel = c("normal"),
    bandwidth = 0.5)
```

```
# Make plot
plot(auto.ts, col = "gray", ylab = "Auto Sales (1000s)",
    main = "Auto Sales - Kernel Smoothed")
lines(auto.k.smooth.wide$x, auto.k.smooth.wide$y, col = "red")
lines(auto.k.smooth.narrow$x, auto.k.smooth.narrow$y, col = "blue")
abline(lm(auto.ts ~ time(auto.ts)), lty = "dotted", col = "black")
```




```
norm.unem <- (unem.intersect.ts - mean(unem.intersect.ts))/sd(unem.intersect.ts)
norm.auto <- (auto.intersect.ts - mean(auto.intersect.ts))/sd(auto.intersect.ts)

plot(norm.unem, col = "blue", ylab = "Std. Unemployment and Auto Sales",
    main = "Standardized Unemployment and Auto Sales", ylim = c(-3,
        5))
lines(norm.auto, col = "red")
legend("topleft", col = c("blue", "red"), legend = c("Unemployment",
    "Auto Sales"), lty = c(1, 1))
```

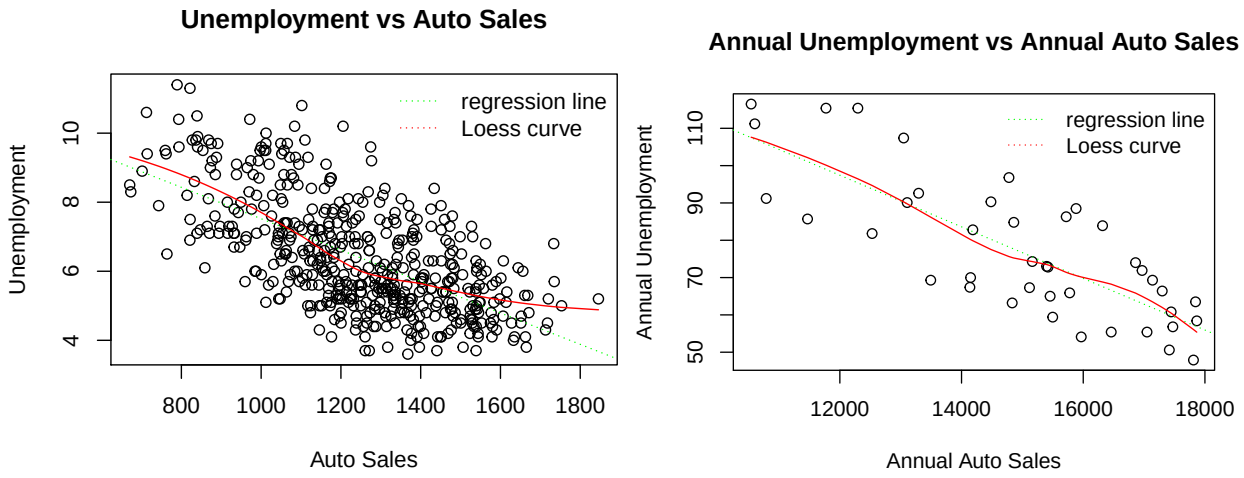


It appears that between 1976 to early 1990s and between late 2000s and 2017, a crest in auto sales would be echoed with a trough in unemployment rate a few months later and a trough in auto sales would be echoed by a crest in unemployment rate a few months later. The unemployment series has a slight downward trend over the whole time interval while the opposite is true for auto sales.

```
# Function for scatterplot with Loess Curve and
# regression curve
scatter.loess.lm.plot = function(y, x, xlab, ylab, title) {

    plot(x = x, y = y, xlab = xlab, ylab = ylab)
    title(title)

    abline(lm(y ~ x), col = "green", lty = "dotted")

    order.pred = order(x)
    smooth.stand = loess(formula = y ~ x, weights = rep(1,
        length(x)))
    lines(x = x[order.pred], y = predict(smooth.stand)[order.pred],
        lty = "solid", col = "red")

    legend("topright", legend = c("regression line", "Loess curve"),
        col = c("green", "red"), lty = "dotted", "solid",
        bty = "n")
}
```

```
scatter.loess.lm.plot(y = unem.intersect.ts, x = auto.intersect.ts,
    xlab = "Auto Sales", ylab = "Unemployment", title = "Unemployment vs Auto Sales")

scatter.loess.lm.plot(y = as.vector(aggregate(unem.intersect.ts)),
    x = as.vector(aggregate(auto.intersect.ts)), xlab = "Annual Auto Sales",
```

```
    ylab = "Annual Unemployment", title = "Annual Unemployment vs Annual Auto Sales")
```



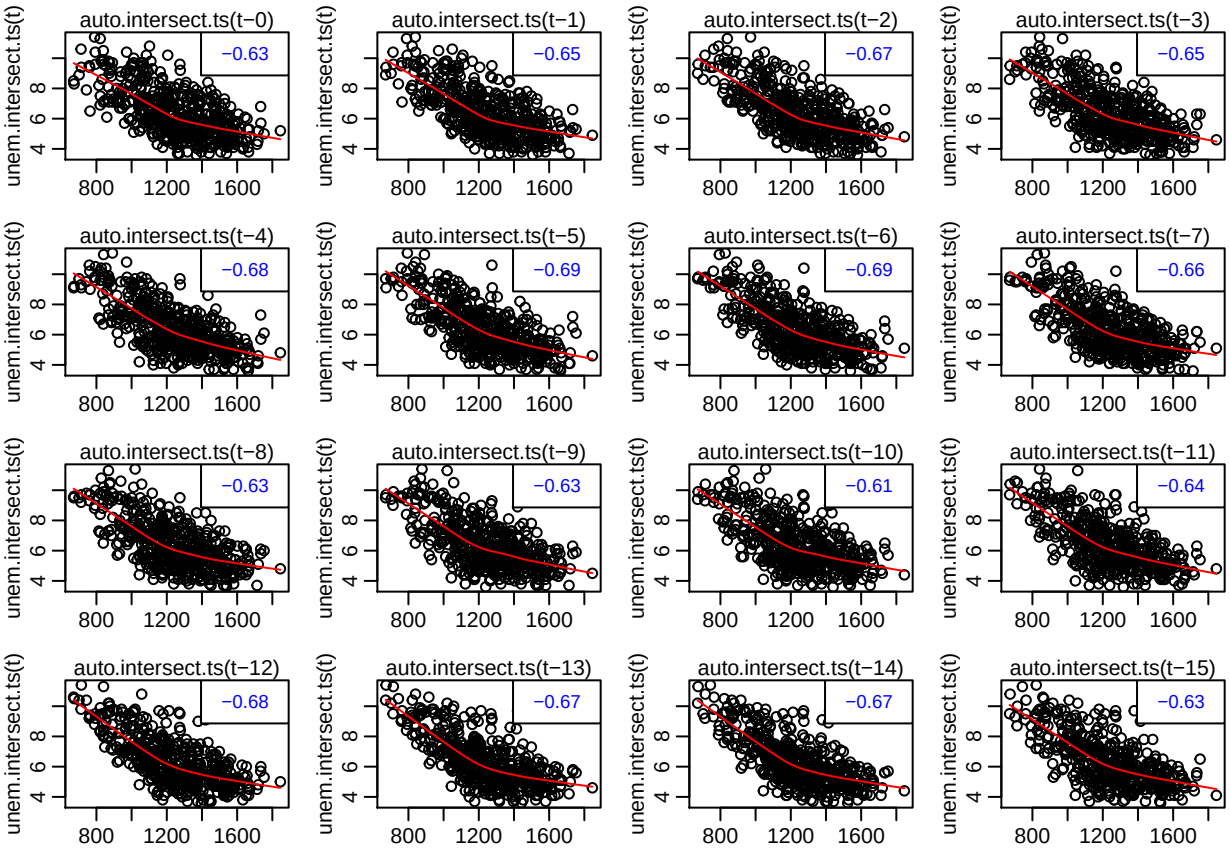

```
corr.unemp.auto <- cor(auto.intersect.ts, unem.intersect.ts)
corr.unemp.auto.ann <- cor(aggregate(auto.intersect.ts),
    aggregate(unem.intersect.ts))
```

Corr(Unemployment, Auto Sales): -0.634

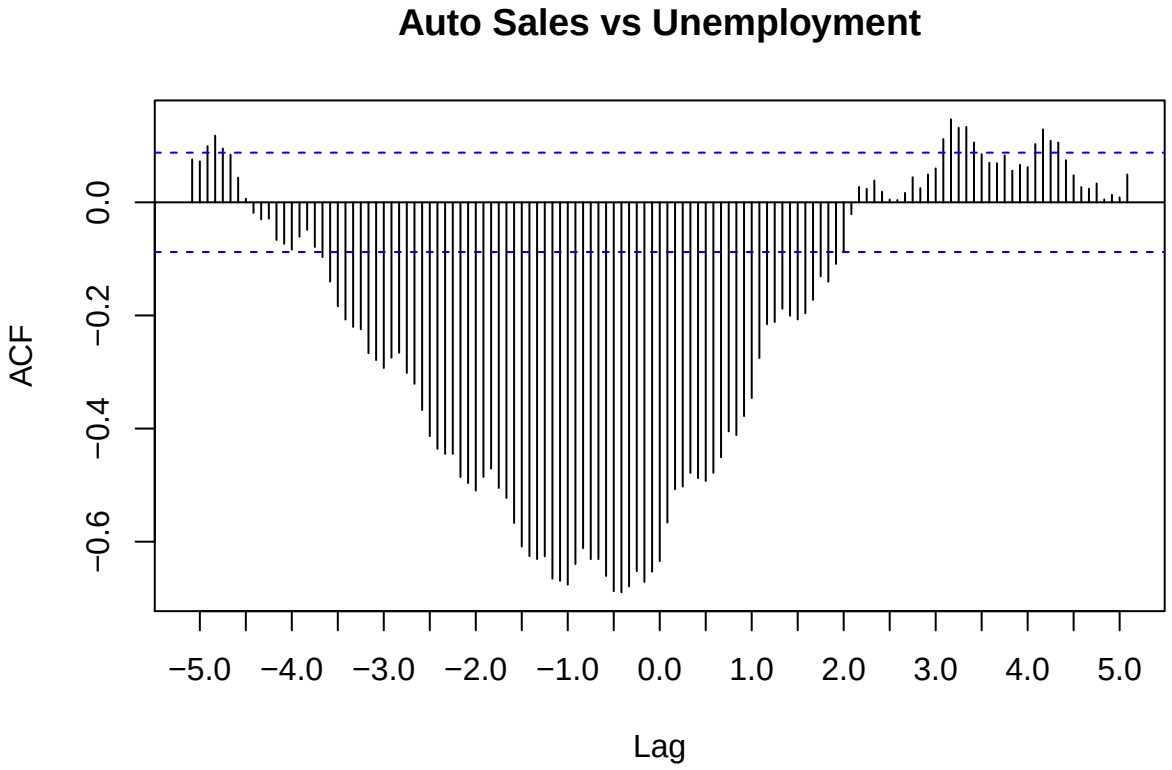
Corr(Annual Unemployment, Annual Auto Sales): -0.789

Disregarding time-dependent variations, the two series show some non-linear relationship in addition to overall negative correlation. The annually aggregated series show stronger and more linear correlation. It seems that elimination of seasonal granularity may "hide" otherwise non-linear relationships, which can be depicted by the scatterplot matrix below. Correlation of unemployment series against auto sales series remains moderate for more than 12 lags, and it peaks at lag 5 and 6.

```
astsa::lag2.plot(auto.intersect.ts, unem.intersect.ts, 15)
```




```
ccf(auto.intersect.ts, unem.intersect.ts, main = "", lag.max = 61,
    xaxt = "n")
axis(side = 1, at = seq(-5, 5, 0.5))
title("Auto Sales vs Unemployment")
```



The ccf plot above assess the cross-correlation $\hat{\rho}_{xy}(h) = \frac{\hat{\gamma}_{xy}(h)}{\sqrt{\hat{\gamma}_x(0)\hat{\gamma}_y(0)}}$ based on the cross-covariance function $\hat{\gamma}_{xy}(h) = n^{-1}\sum_{t=1}^{n-h}(x_{t+h} - \bar{x})(y_t - \bar{y})$, where x_t refers to the auto sales series, y_t refers to the unemployment series, h refers to the number of lags in the auto sales series and n refers to the number of monthly observations considered. The ccf occurs mostly at the negative lags and peaks somewhere at lag 5 to 6, this indicate that auto sales series clearly leads the unemployment series.

Testing for Cointegration

Based on the unit root test results conducted in the univariate section and the moderate, negative linear correlation observed just above. There is some chance that our two series are cointegrated. We conduct the Phillips-Ouliaris Cointegration Test here with the null hypothesis:

- Ho: The two series are not cointegrated
- Ha: the two series are cointegrated

```
tseries::po.test(cbind(auto.intersect.ts, unem.intersect.ts))
```

```
## Warning in tseries::po.test(cbind(auto.intersect.ts, unem.intersect.ts)):
## p-value smaller than printed p-value
```

```
##
##  Phillips-Ouliaris Cointegration Test
##
## data:  cbind(auto.intersect.ts, unem.intersect.ts)
## Phillips-Ouliaris demeaned = -234.14, Truncation lag parameter =
## 4, p-value = 0.01
```

The po.test results provides evidence that the series are cointegrated since the null hypothesis is rejected at the 1% level.

If there truely exists a linear combination between the two series that is stationary, regressing unemployment

series on auto sales should produce residuals that are some what stationary (though it can still be time dependent). We conduct such a model below to examine the residual series.

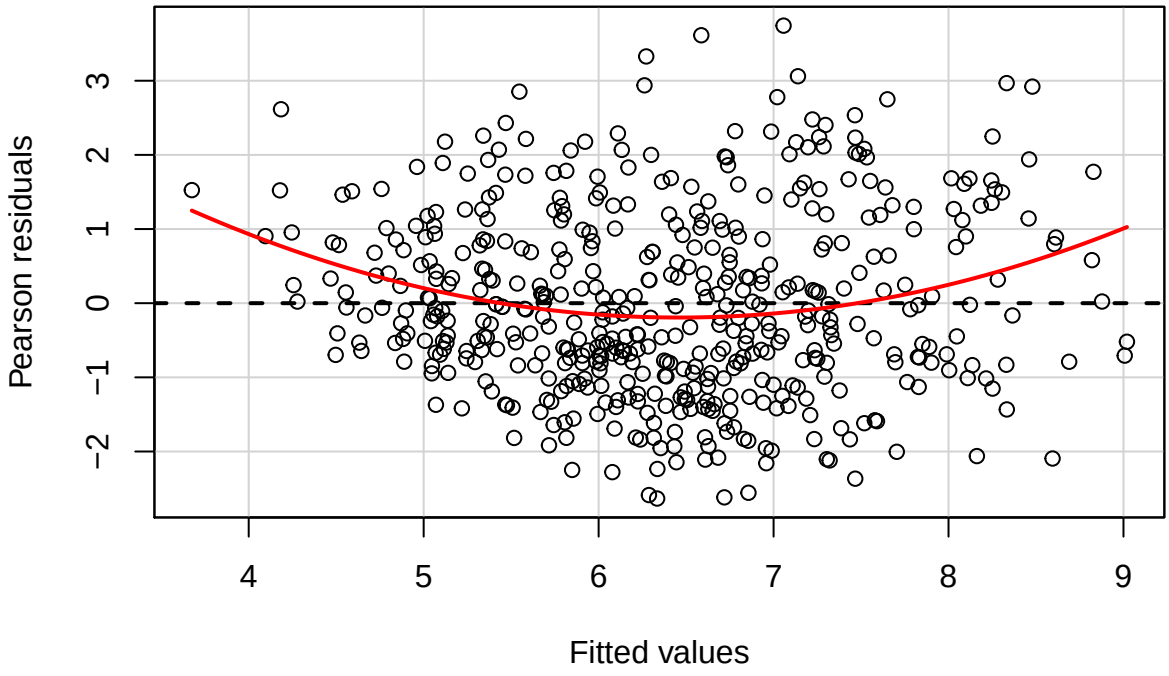
```
unem.auto.lm = lm(unem.intersect.ts ~ auto.intersect.ts)
# summary(unem.auto.lm)
unem.auto.lm$coefficients[2]
```

```
## auto.intersect.ts
##      -0.004547657
```

```
as.numeric(unem.auto.lm$coefficients[2]/sd(unem.intersect.ts))
```

```
## [1] -0.002833825
```

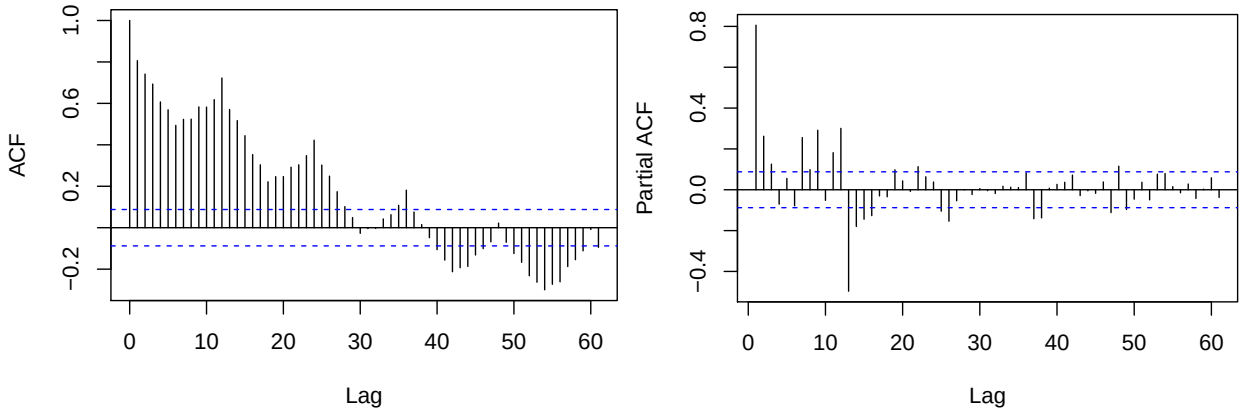
```
car::residualPlot(unem.auto.lm)
```



From the regression output above, the coefficient estimate for auto sales is not practically significant(less than 1% standard deviation of unemployment rate). The residual plot also shows strong curvature. Auto sales at lag 0 don't effectively explain variations in unemployment in expectation.

```
par(mfrow = c(2, 1))
ts.plot(unem.intersect.ts, main = "", ylab = "Unemployment Rate")
title("Unemployment Rate")
unem.auto.lm.res = resid(unem.auto.lm)
plot(unem.auto.lm.res, xlab = "t", ylab = "Residuals", lty = 1,
    pch = 1, type = "l")
title("Residual Series Unemployment ~ Auto Sales")
```


```
acf(unem.auto.lm.res, 61, main = "")
pacf(unem.auto.lm.res, 61, main = "")
```



From the time plots and acf plots, the residual series still picks up most of the random walk behaviors in the unemployment rate. The acf and pacf plots still show evidence of a strong AR(1) process. Contradicting the cointegration test results, these residual plots don't fully support the existence of linear combination between the two series that is entirely stationary.

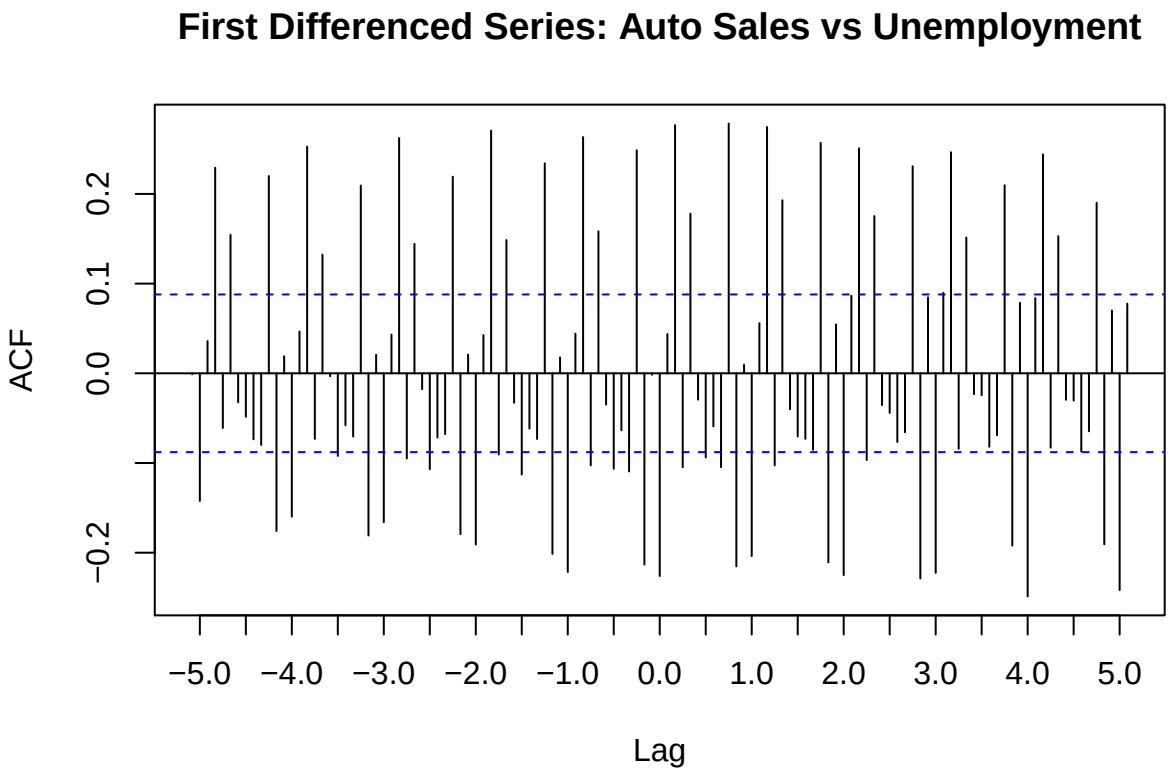
There is clearly a linear relationship between the unemployment and auto sales, but the two are not necessarily cointegrated for models like VECM to be valid. Our poor OLS residual behavior also demonstrated that OLS is not appropriate, on top of the fact that such a model cannot do forecast on any given time point.

Establish Stationarity for VAR models

An alternative is to construct a VAR model which requires stationary input series. Our unit root test provided strong evidence of unit root in the unemployment series and weak evidence for that in the auto sales series,

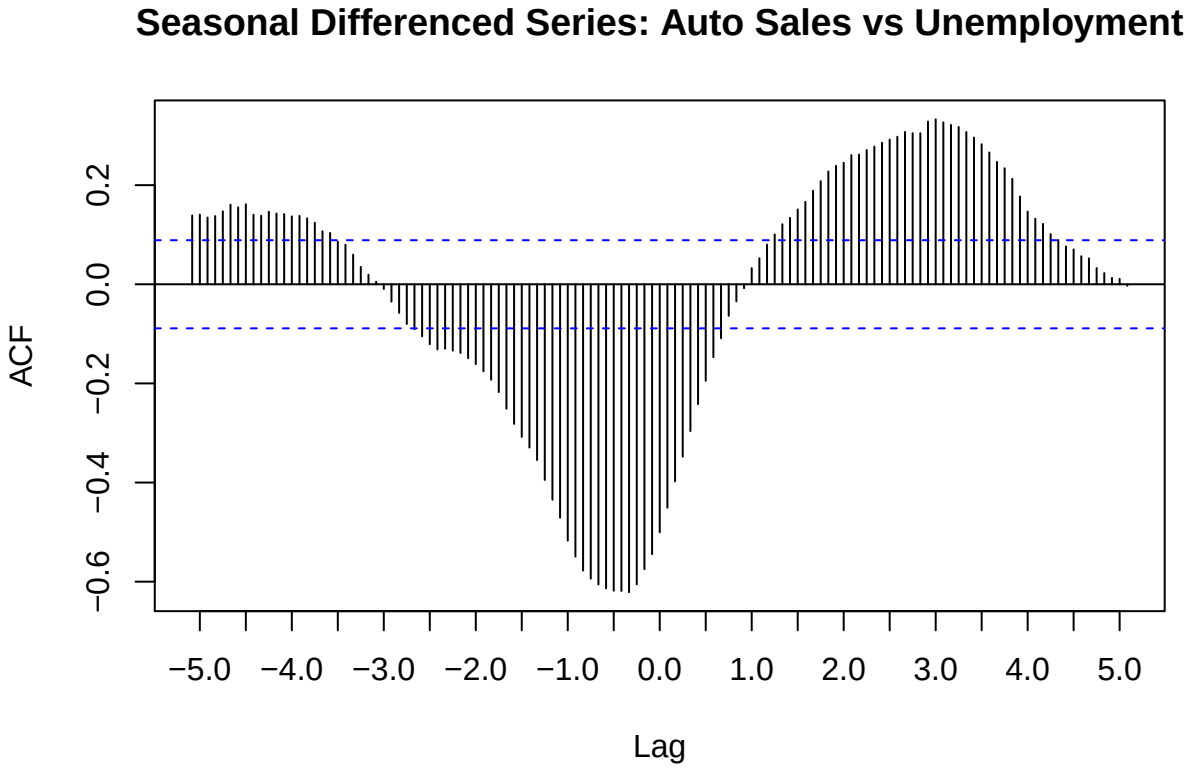
and our univariate plots exhibit noticeable seasonal patterns. We proceed to examine the cross-correlation plots between the differenced series.

```
combined.diff = ts.intersect(unem.ts.diff, auto.ts.diff)
unem.intersect.ts.diff = combined.diff[, 1]
auto.intersect.ts.diff = combined.diff[, 2]

# head(combined.diff);tail(combined.diff)

ccf(auto.intersect.ts.diff, unem.intersect.ts.diff, main = "",
    lag.max = 61, xaxt = "n")
axis(side = 1, at = seq(-5, 5, 0.5))
title("First Differenced Series: Auto Sales vs Unemployment")
```



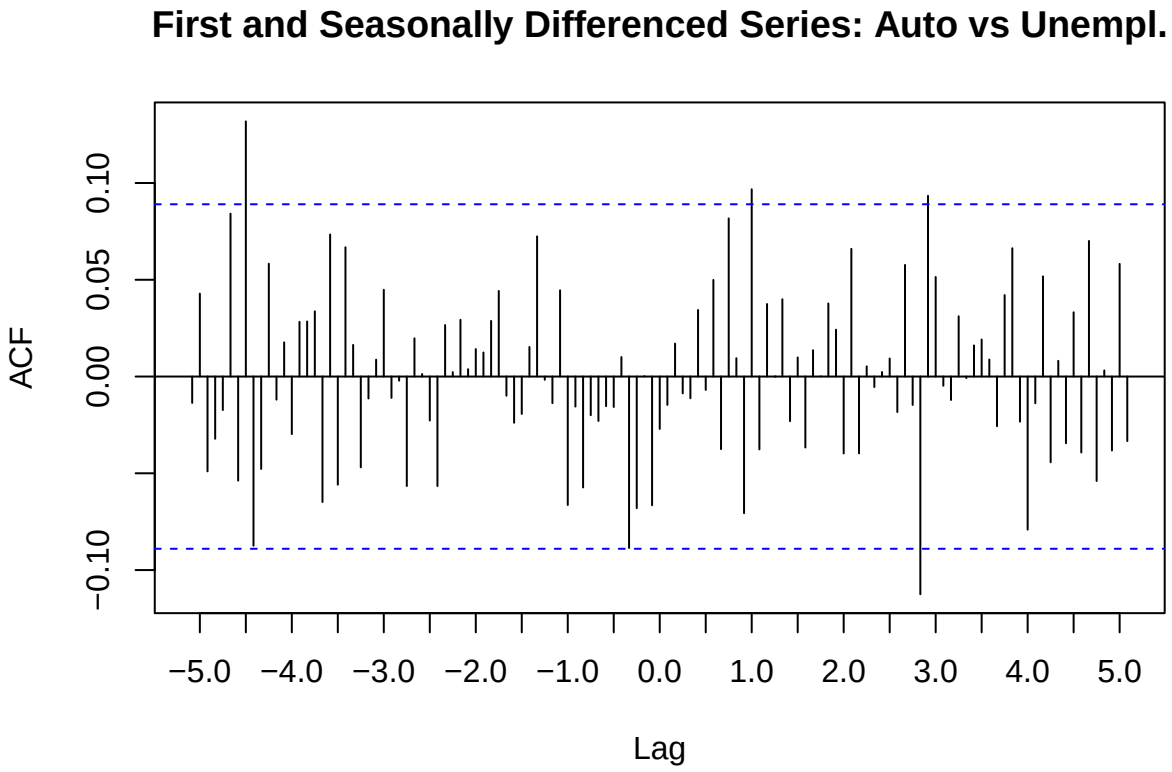
Cross-correlation for the first differenced series shows obvious significant values around lag 12, 24, 36, 48, and so on. This says that our VAR model can be more effective if the series are differenced, or accounted by seasonal variables.

```
combined.diff12 = ts.intersect(unem.ts.diff12, auto.ts.diff12)
unem.intersect.ts.diff12 = combined.diff12[, 1]
auto.intersect.ts.diff12 = combined.diff12[, 2]

ccf(auto.intersect.ts.diff12, unem.intersect.ts.diff12,
    main = "", lag.max = 61, xaxt = "n")
axis(side = 1, at = seq(-5, 5, 0.5))
title("Seasonal Differenced Series: Auto Sales vs Unemployment")
```



Cross-correlation for the seasonally differenced series shows obvious gradual decay around lag 3-7 and 36, which can be the effect of the inherent unit roots. This suggests that we should try first differencing the two series for the VAR model.

```
combined.diff.diff12 = ts.intersect(unem.ts.diff.diff12,
    auto.ts.diff.diff12)
unem.intersect.ts.diff.diff12 = combined.diff.diff12[, 1]
auto.intersect.ts.diff.diff12 = combined.diff.diff12[, 2]


ccf(auto.intersect.ts.diff.diff12, unem.intersect.ts.diff.diff12,
    main = "", lag.max = 61, xaxt = "n")
axis(side = 1, at = seq(-5, 5, 0.5))
title("First and Seasonally Differenced Series: Auto vs Unempl.")
```



The cross-correlation for the first and seasonally differenced series show slightly significance at lag -54 and lag 34.

The above cross-correlation study and earlier unit root tests suggest that we should consider the following as input for the VAR models:

- Both series first differenced and seasonal variable. Lags undetermined.
- Both series seasonally differenced and include 4:6 lags
- Both series first and seasonally differenced. Lags undetermined.

Model Specification

```
# split into train and test
unem.train = unem.data[0:816, ]
unem.test = unem.data[817:834, ]
unem.train.ts = ts(unem.train$UNRATENSA)
unem.test.ts = ts(unem.test$UNRATENSA)
```

Construct and Loop Through Search Spaces

From the EDA, we speculated the following models:

- Seasonal Differenced Series: SARIMA(4,0,1)(3,1,Q)
- First Differenced Series: SARIMA(9,1,q)(1,0,1)
- First and Seasonal Differenced Series: SARIMA(4,1,0)(0,1,1)

Because the EDA was inconclusive on differencing strategies, we define the following search spaces:

- Seasonal Differenced Series: SARIMA(1:4,0,0:3)(0:4,1,0:3)

- First Differenced Series: SARIMA(1:9,1,0:3)(0:3,0,0:3)
- First and Seasonal Differenced Series: SARIMA(0:5,1,0:3)(0:3,1,0:3)

where SARIMA(p,d,q)(P,D,Q)_12 is the general form of model.

Since we cannot directly compare the AICs of models with different orders of differencing, we explore each of these separately. We will identify the models from each search space with the best AIC, then examine the behavior of the residuals and the out of sample fit in order to compare across different orders of differencing.

```
# READ LOOP RESULTS FROM THE CSV FILES

# Seasonal Differenced Series :
# SARIMA(1:4,0,0:3)(0:4,1,0:3)
bestAIC <- 10000
unem.diff12.df = data.frame(p = 0, q = 0, P = 0, Q = 0,
    aic = bestAIC)

for (p in 1:4) {
    for (q in 0:3) {
        for (P in 0:4) {
            for (Q in 0:3) {
                cat(p, q, P, Q, "\n")
                try(m <- Arima(unem.train.ts, order = c(p,
                  0, q), seasonal = list(order = c(P, 1,
                  Q), period = 12)))

                if (m$aic < bestAIC)
                  # update if this model attain better aic
                {
                  bestAIC = m$aic
                  bestFit = m
                  bestModel = c(p, q, P, Q)
                  cat(p, q, P, Q, as.numeric(bestAIC), "\n")
                  unem.diff12.df = rbind(unem.diff12.df,
                    data.frame(p = p, q = q, P = P, Q = Q,
                      aic = bestAIC))
                }
            }
        }
    }
}

unem.diff12.df = unem.diff12.df[seq(dim(unem.diff12.df)[1],
    1), ]
write.csv(x = unem.diff12.df, file = "unem.diff12.df.csv")

# First Differenced Series: SARIMA(1:9,1,0:3)(0:3,0,0:3)

bestAIC <- 10000

unem.diff.df = data.frame(p = 0, q = 0, P = 0, Q = 0, aic = bestAIC)

for (p in 1:9) {
    for (q in 0:3) {
        for (P in 0:3) {
```

```
            for (Q in 0:3) {
                cat(p, q, P, Q, "\n")
                try(m <- Arima(unem.train.ts, order = c(p,
                  1, q), seasonal = list(order = c(P, 0,
                  Q), period = 12)), silent = TRUE)

                if (m$aic < bestAIC)
                  # update if this model attain better aic
                {
                  bestAIC = m$aic
                  bestFit = m
                  bestModel = c(p, q, P, Q)
                  cat(p, q, P, Q, as.numeric(bestAIC), "\n")
                  unem.diff.df = rbind(unem.diff.df, data.frame(p = p,
                    q = q, P = P, Q = Q, aic = bestAIC))
                }
            }
        }
    }
}

unem.diff.df = unem.diff.df[seq(dim(unem.diff.df)[1], 1),
    ]
write.csv(x = unem.diff.df, file = "unem.diff.df.csv")

# First and Seasonal Differenced Series:
# SARIMA(0:5,1,0:3)(0:3,1,0:3)

bestAIC <- 10000

unem.diff.diff12.df = data.frame(p = 0, q = 0, P = 0, Q = 0,
    aic = bestAIC)

for (p in 0:5) {
    for (q in 0:3) {
        for (P in 0:3) {
            for (Q in 0:3) {
                cat(p, q, P, Q, "\n")
                try(m <- Arima(unem.train.ts, order = c(p,
                  1, q), seasonal = list(order = c(P, 1,
                  Q), period = 12)), silent = TRUE)

                if (m$aic < bestAIC)
                  # update if this model attain better aic
                {
                  bestAIC = m$aic
                  bestFit = m
                  bestModel = c(p, q, P, Q)
                  cat(p, q, P, Q, as.numeric(bestAIC), "\n")
                  unem.diff.diff12.df = rbind(unem.diff.diff12.df,
                    data.frame(p = p, q = q, P = P, Q = Q,
                      aic = bestAIC))
                }
```

```
            }
        }
    }
}

unem.diff.diff12.df = unem.diff.diff12.df[seq(dim(unem.diff.diff12.df)[1],
    1), ]
write.csv(x = unem.diff.diff12.df, file = "unem.diff.diff12.df.csv")
```

Seasonal Differenced Search Result

```
unem.diff12.df = read.csv("unem.diff12.df.csv")
cat("Top candidates for seasonal differenced model: d = 0, D = 1 \n")
```

```
## Top candidates for seasonal differenced model: d = 0, D = 1
```

```
unem.diff12.df$AIC <- round(unem.diff12.df$aic, 3)
head(unem.diff12.df[, c(3:6, 8)], 8)
```

```
##   p q P Q     AIC
## 1 4 3 0 1 -37.666
## 2 3 1 0 1 -36.061
## 3 2 3 2 3 -35.620
## 4 2 2 3 3 -35.614
## 5 2 2 0 1 -35.562
## 6 2 1 3 3 -34.077
## 7 2 1 0 1 -32.641
## 8 1 3 2 3 -11.354
```

For the Seasonally Differenced search space, the model with the best AIC is a SARIMA(4,0,3)(0,1,1)[12]. We will further evaluate this model in later sections.

First Differenced Search Result

```
unem.diff.df = read.csv("unem.diff.df.csv")
cat("Top candidates for first differenced model: d = 1, D = 0 \n")
```

```
## Top candidates for first differenced model: d = 1, D = 0
```

```
unem.diff.df$AIC <- round(unem.diff.df$aic, 3)
head(unem.diff.df[, c(3:6, 8)], 8)
```

```
##   p q P Q     AIC
## 1 8 3 1 1 -30.466
## 2 2 3 2 1 -21.842
## 3 2 1 1 1  -4.512
## 4 1 2 1 1  -4.141
## 5 1 1 3 3  -1.156
## 6 1 1 1 1  -0.009
## 7 1 0 1 1  36.078
## 8 1 0 1 0 327.916
```

For the First Differenced search space, the model with the best AIC is a SARIMA(8,1,3)(1,0,1)[12]. We will further evaluate this model in later sections.

First and Seasonal Differenced Search Result

```
unem.diff.diff12.df = read.csv("unem.diff.diff12.df.csv")
cat("Top candidates for first and seasonal differenced model: d = 1, D = 1 \n")
```

```
## Top candidates for first and seasonal differenced model: d = 1, D = 1
```

```
unem.diff.diff12.df$AIC <- round(unem.diff.diff12.df$aic,
    3)
head(unem.diff.diff12.df[, c(3:6, 8)], 8)
```

```
##   p q P Q     AIC
## 1 2 3 1 1 -28.399
## 2 2 1 0 1 -18.350
## 3 1 2 3 3 -18.275
## 4 1 2 0 1 -18.019
## 5 1 1 3 3 -14.603
## 6 1 1 0 1 -12.696
## 7 0 3 2 3  -4.160
## 8 0 3 0 1  -2.766
```

For the First and Seasonally Differenced search space, the model with the best AIC is a SARIMA(2,1,3)(1,1,1)[12]. We will further evaluate this model in later sections.

Using the results above, we attempt to simplify the order of the top candidate for each search space by comparing their residuals against respective lower order models.

Seasonal Differenced Models (d = 0 , D = 1)

```
# top (4,0,3)(0,1,1)
m.diff12.1 <- Arima(unem.train.ts, order = c(4, 0, 3), seasonal = list(order = c(0,
    1, 1), period = 12))

# 2nd (3,0,1)(0,1,1)
m.diff12.2 <- Arima(unem.train.ts, order = c(3, 0, 1), seasonal = list(order = c(0,
    1, 1), period = 12))

# 7th (2,0,1)(0,1,1)
m.diff12.7 <- Arima(unem.train.ts, order = c(2, 0, 1), seasonal = list(order = c(0,
    1, 1), period = 12))
```

```
# Function to print residual charts
print_resid_chart <- function(mod) {
    cat("Model SARIMA ", c(mod$arma[1], mod$arma[6], mod$arma[2],
        mod$arma[3], mod$arma[7], mod$arma[4]), ":\n")
    cat("AIC: ", (mod$aic), "\n")
    cat("BIC: ", (mod$bic), "\n")
    qqnorm(mod$residuals, main = "Normal Q-Q Plot of Residuals")
    hist(mod$residuals, main = "Histogram of Residuals",
        xlab = "Residuals")
    acf <- acf(mod$residuals, 100, plot = FALSE)
    pacf <- pacf(mod$residuals, 100, plot = FALSE)
    plot(acf, main = "ACF of Residuals")
    plot(pacf, main = "PACF of Residuals")
```

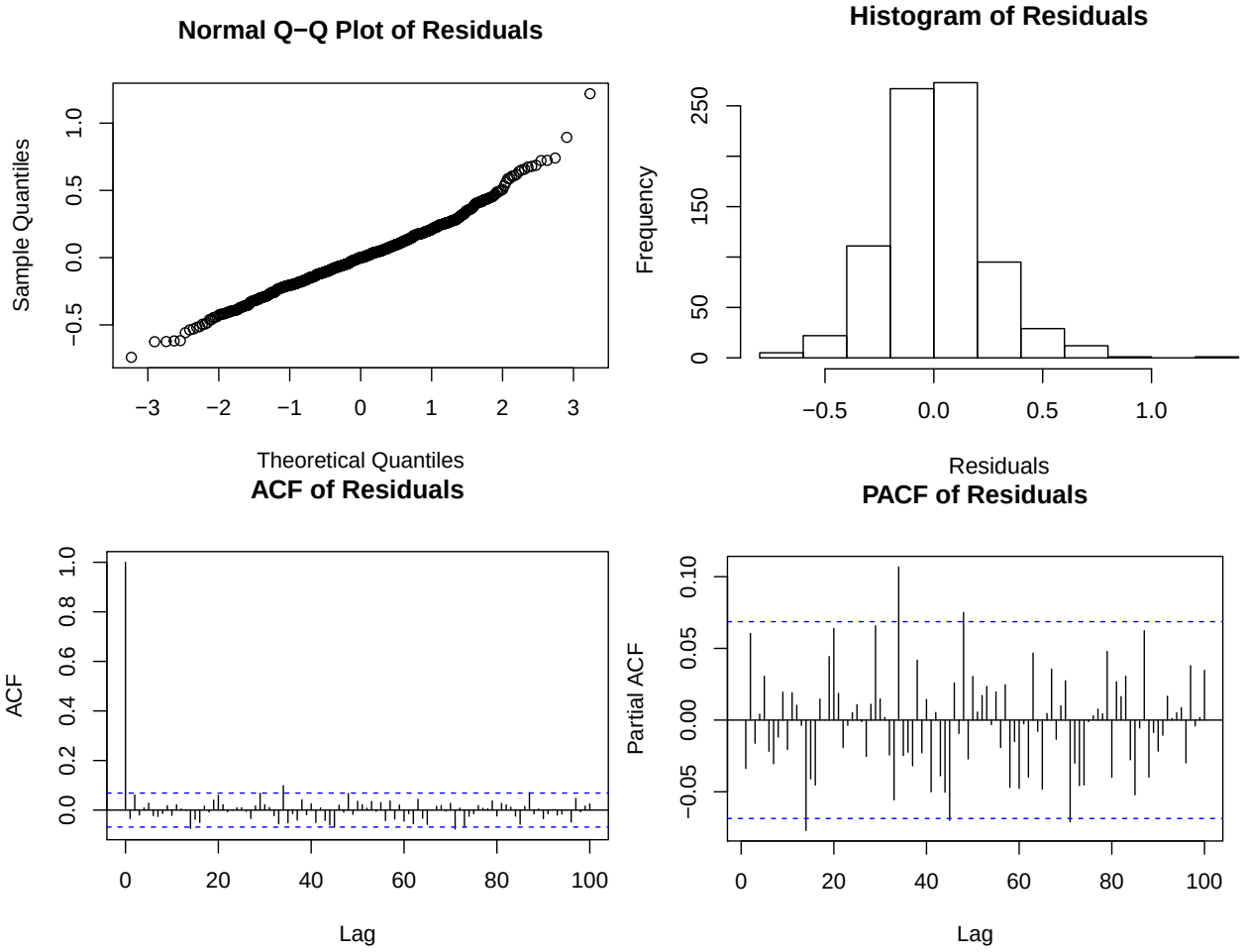
```
}
```

```
# top (4,0,3)(0,1,1)
print_resid_chart(m.diff12.1)
```

```
## Model SARIMA  4 0 3 0 1 1 :
## AIC:  -37.63256
## BIC:  4.573828
```

```
# top (4,0,3)(0,1,1)
Box.test(m.diff12.1$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff12.1$residuals
## X-squared = 20.237, df = 25, p-value = 0.7343
```



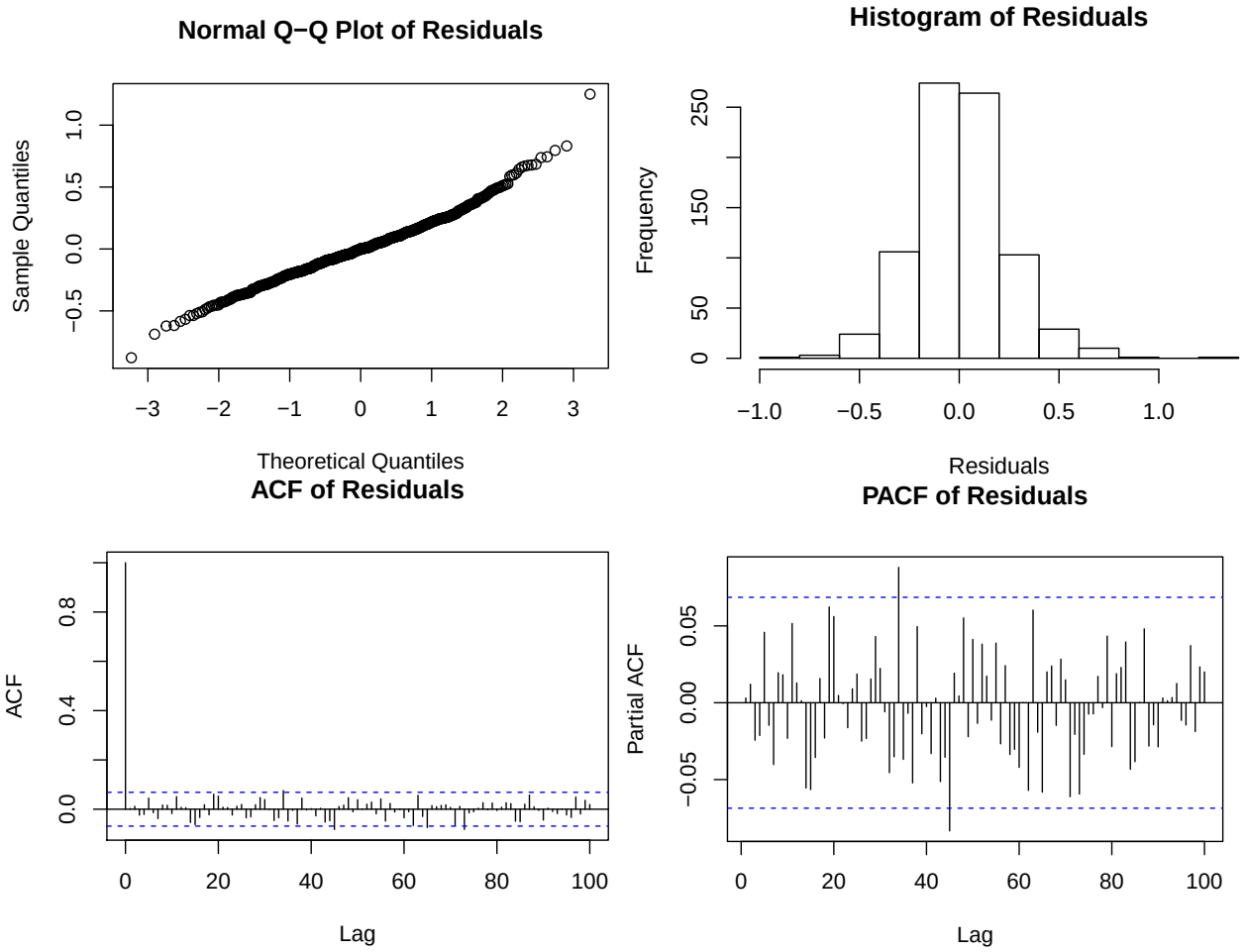
Examining the residuals from our first candidate model SARIMA(4,0,3)(0,1,1)[12], we note from the histogram and Normal Q-Q Plot that the residuals are not quite normally distributed, since there is one extreme outlier on the right side of the plot. The ACF plot has a few spikes that are nearly significant, but nothing alarming. The PACF plot does show some spikes that are significant, indicating that the residuals may not perfectly resemble white noise. However, the Ljung-Box test fails to reject the null hypothesis that the residuals are independently distributed.

```
# 2nd (3,0,1)(0,1,1)
print_resid_chart(m.diff12.2)
```

```
## Model SARIMA  3 0 1 0 1 1 :
## AIC:  -36.06063
## BIC:  -7.923031
```

```
# 2nd (3,0,1)(0,1,1)
Box.test(m.diff12.2$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff12.2$residuals
## X-squared = 20.778, df = 25, p-value = 0.7049
```



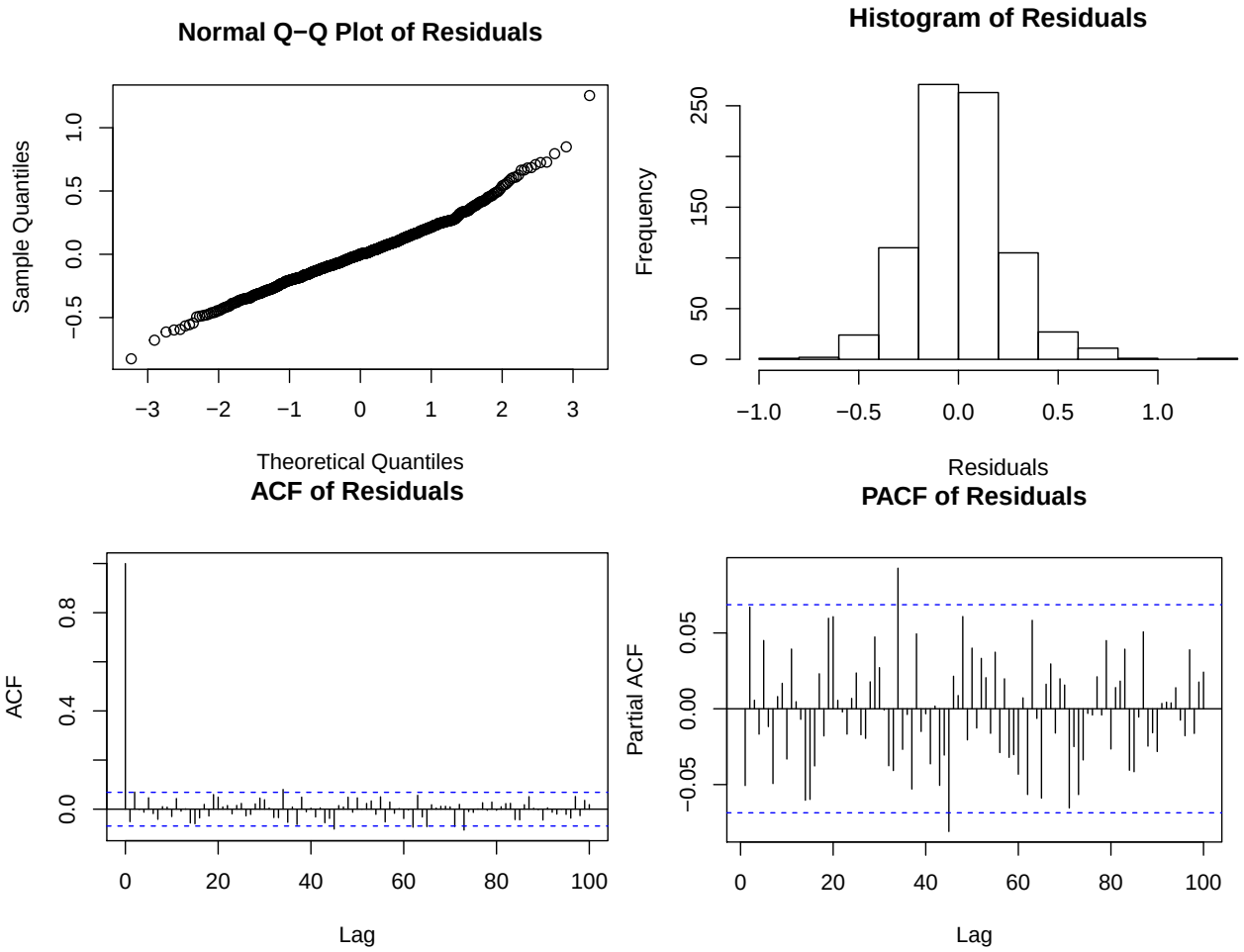
The residuals for the simpler model SARIMA(3,0,1)(0,1,1)[12] are very similar to the first model, although the PACF plot has fewer spikes that are near significance.

```
# 7th (2,0,1)(0,1,1)
print_resid_chart(m.diff12.7)
```

```
## Model SARIMA  2 0 1 0 1 1 :
## AIC:  -32.64114
## BIC:  -9.193147
```

```r
# 7th (2,0,1)(0,1,1)
Box.test(m.diff12.7$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff12.7$residuals
## X-squared = 25.836, df = 25, p-value = 0.4164
```



Simplifying even further to a SARIMA(2,0,1)(0,1,1)[12], which was the 7th candidate based on the AIC, we note that the residual diagnostics are very similar to the previous model.

For seasonally differenced models, we simplified from the top AIC candidate, SARIMA(4,0,3)(0,1,1)[12] to the 7th candidate SARIMA(2,0,1)(0,1,1)[12] and retain or even improved white noise residual behavior. We will keep these two candidates to compare out of sample performance:

- `SARIMA(4,0,3)(0,1,1)[12]`
- `SARIMA(2,0,1)(0,1,1)[12]`

We also performed the Ljung-Box test above for our residual series to see if residuals for these models are independently distributed. Test hypothesis is as follows:

- Ho : The residuals are independently distributed
- Ha : The residuals are not independently distributed

The tests fail to reject the null hypothesis, thus supporting that the residuals resemble white noise.

First Differenced Models (d = 1 , D = 0)

Next we will fit several first differenced models and examine residual diagnostics similar to our process above.

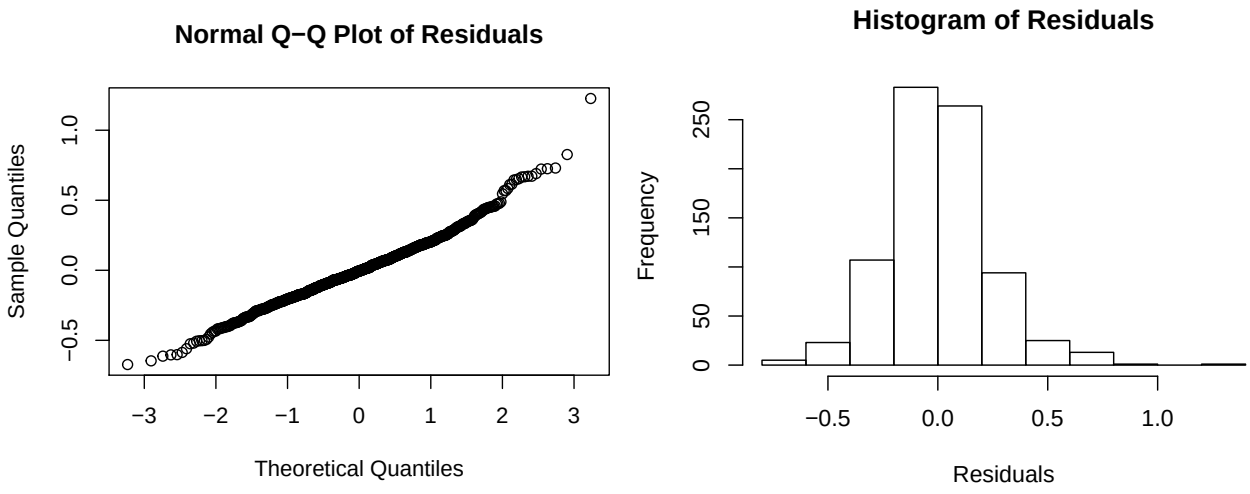
```
# top (8,1,3)(1,0,1)
m.diff.1 <- Arima(unem.train.ts, order = c(8, 1, 3), seasonal = list(order = c(1,
    0, 1), period = 12))

# 3rd (2,1,1)(1,0,1)
m.diff.3 <- Arima(unem.train.ts, order = c(2, 1, 1), seasonal = list(order = c(1,
    0, 1), period = 12))

# 6th (1,1,1)(1,0,1)
m.diff.6 <- Arima(unem.train.ts, order = c(1, 1, 1), seasonal = list(order = c(1,
    0, 1), period = 12))

# 7th (1,1,0)(1,0,1)
m.diff.7 <- Arima(unem.train.ts, order = c(1, 1, 0), seasonal = list(order = c(1,
    0, 1), period = 12))
```

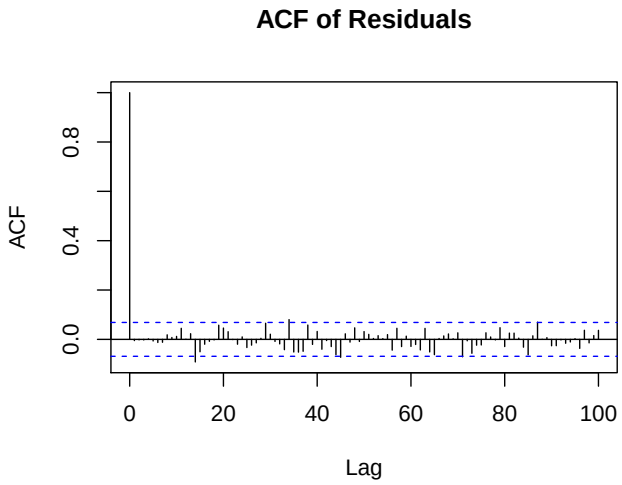
```
# top (8,1,3)(1,0,1)
print_resid_chart(m.diff.1)
```

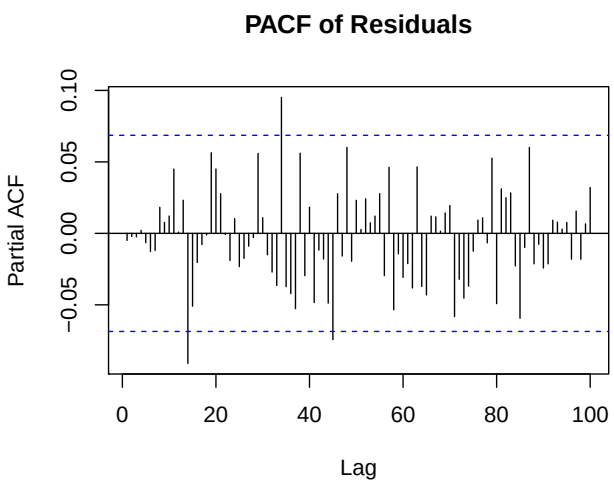
```
## Model SARIMA  8 1 3 1 0 1 :
## AIC:  -30.46596
## BIC:  35.37868
```

```
# top (8,1,3)(1,0,1)
Box.test(m.diff.1$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff.1$residuals
## X-squared = 18.978, df = 25, p-value = 0.7982
```







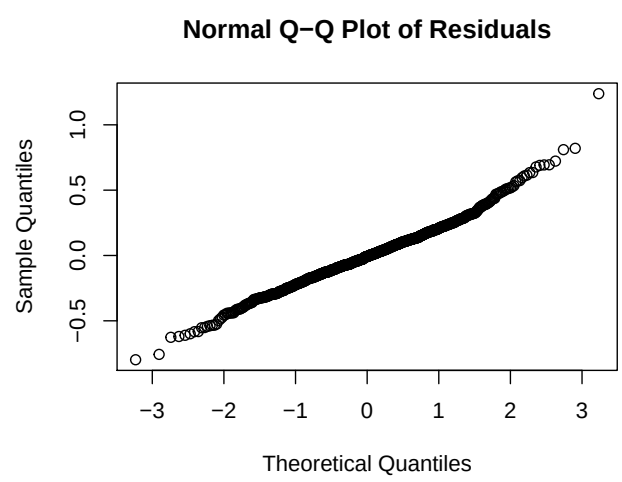
First, we start with the model that had the best AIC within the First Differenced search space, a SARIMA(8,1,3)(1,0,1)[12]. Again, the residuals are approximately normally distributed, with the exception of one outlier. The ACF plot does not have anly particularly significant spikes. The PACF plot does have a few significant spikes. However, since we have 100 lags showing on this plot, 5 of them may be significant just by chance, and this is not too concerning. The Ljung-Box test fails to reject the null hypothesis that the residuals are independently distributed, further supporting our characterization of the residuals as white noise.

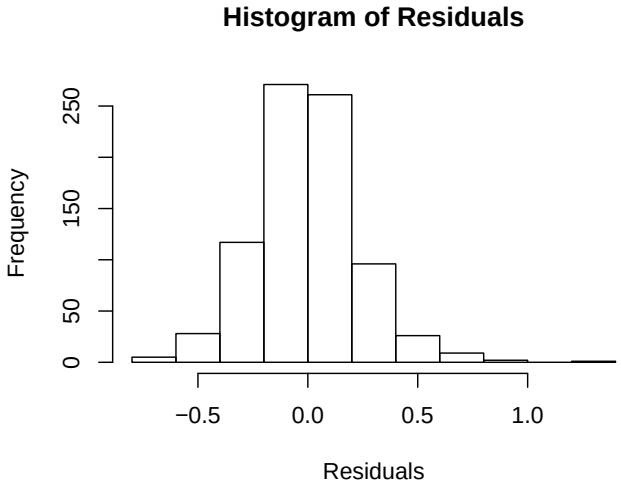
```
# third (2,1,1)(1,0,1)
print_resid_chart(m.diff.3)
```

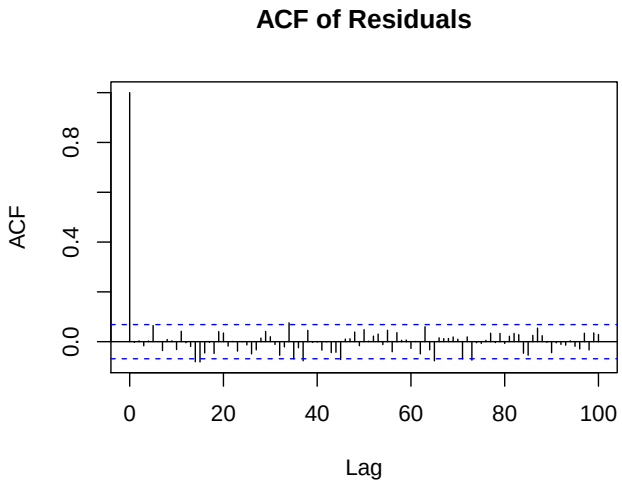
```
## Model SARIMA  2 1 1 1 0 1 :
## AIC:  -4.512136
## BIC:  23.70699
```

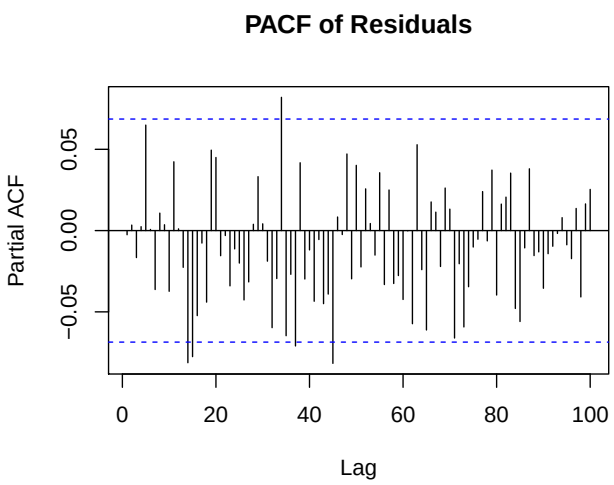
```
# third (2,1,1)(1,0,1)
Box.test(m.diff.3$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff.3$residuals
## X-squared = 25.946, df = 25, p-value = 0.4105
```









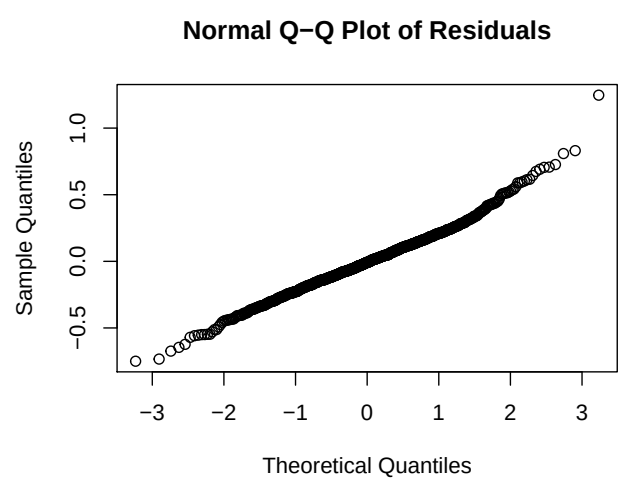
Next, we fit a much simpler model which also had a reasonably low AIC, a SARIMA(2,1,1)(1,0,1)[12]. The residual diagnostics for this model are similar to the previous model, so we have not sacrificed the white noise of the residuals from this simplification.

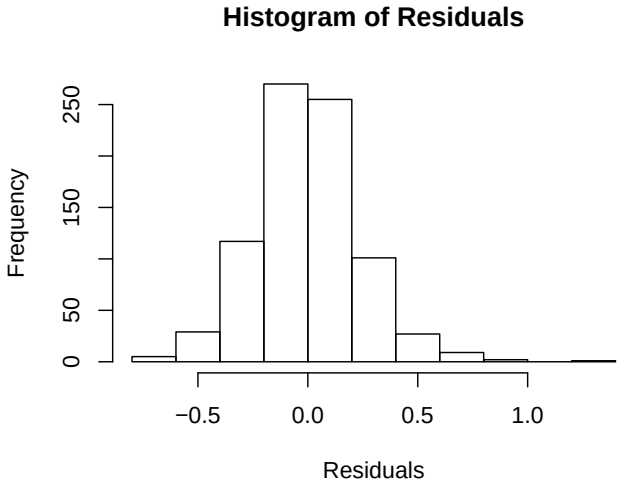
```
# 6th (1,1,1)(1,0,1)
print_resid_chart(m.diff.6)
```

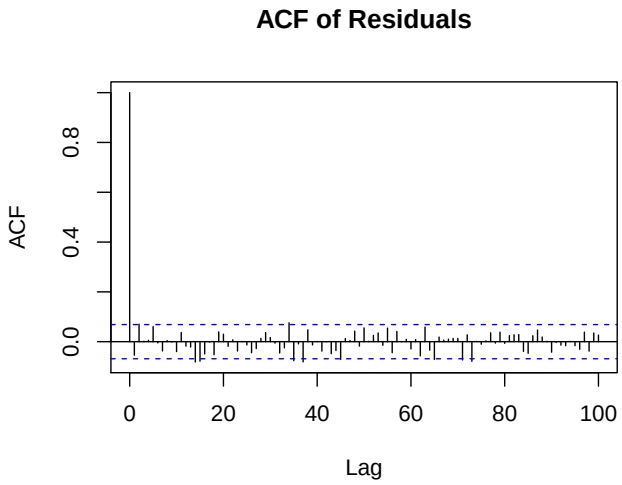
```
## Model SARIMA  1 1 1 1 0 1 :
## AIC:  -0.008886923
## BIC:  23.50705
```

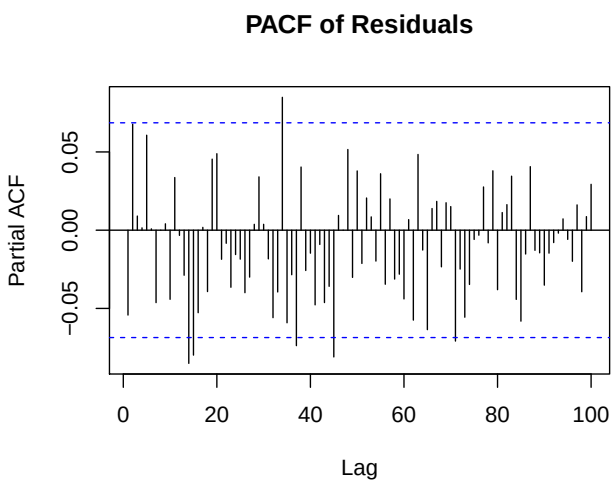
```
# 6th (1,1,1)(1,0,1)
Box.test(m.diff.6$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff.6$residuals
## X-squared = 32.498, df = 25, p-value = 0.1441
```









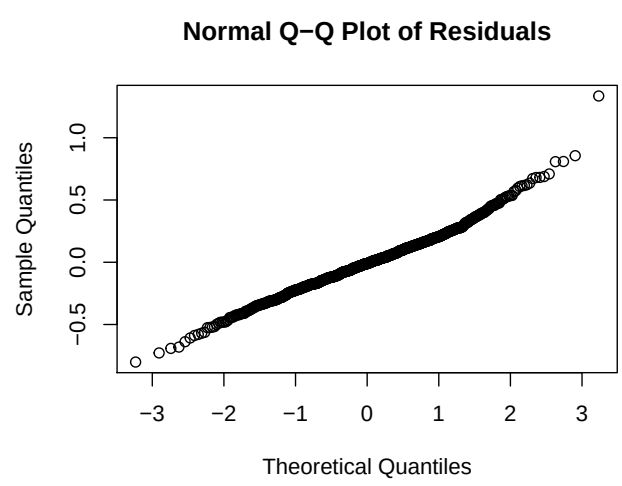
Next, we simplify the model one step further, with a SARIMA(1,1,1)(1,0,1)[12]. However, at this point, the PACF has additional spikes that are significant or nearly significant, and more spikes than would happen by chance. Therefore, this level of simplification may be too extreme. However, the Ljung-Box test still fails to reject the null hypothesis that the residuals are independently distributed.

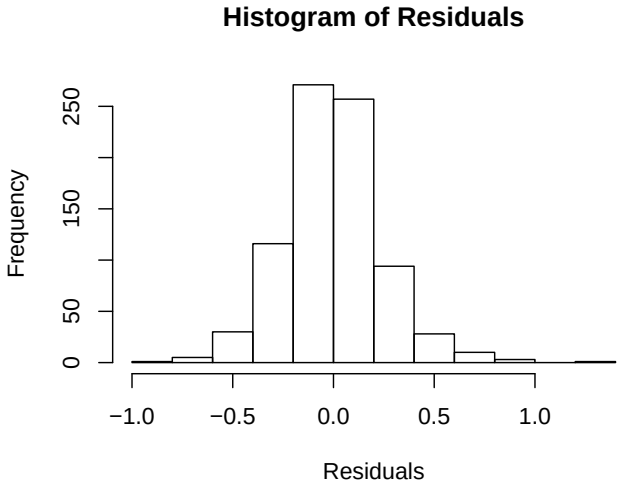
```
# 7th (1,1,0)(1,0,1)
print_resid_chart(m.diff.7)
```

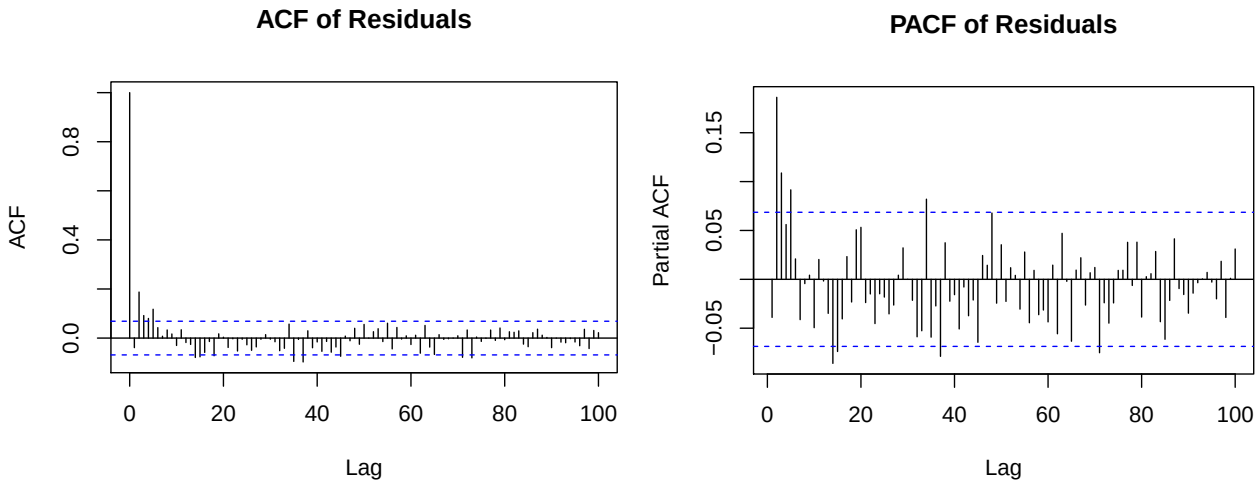
```
## Model SARIMA  1 1 0 1 0 1 :
## AIC:  36.07811
## BIC:  54.89086
```

```
# 7th (1,1,0)(1,0,1)
Box.test(m.diff.7$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff.7$residuals
## X-squared = 80.635, df = 25, p-value = 9.071e-08
```







Once we simplify the model to SARIMA(1,1,0)(1,0,1)[12], the residuals no longer resemble white noise. This can be seen from the PACF plot, where there are very significant spikes at early lags. In this case, the Ljung-Box test strongly rejects the null hypothesis that the residuals are independently distributed.

For first differenced models, we reduced from the top AIC candidate, SARIMA(8,1,3)(1,0,1)[12] to the 3rd candidate SARIMA(2,1,1)(1,0,1)[12] and retain white noise residual behavior. The 6th candidate SARIMA(1,1,1)(1,0,1)[12] is still satisfactory but more of its lags are slightly closer to the cut-off. The 7th candidate SARIMA(1,1,0)(1,0,1)[12] starts to show significant spikes on the PACF plot at lag 2, 3 and 5. Therefore we stop searching from there downwards. We will keep the first three candidates to compare out of sample performance:

- SARIMA(8,1,3)(1,0,1)[12]
- SARIMA(2,1,1)(1,0,1)[12]
- SARIMA(1,1,1)(1,0,1)[12]

The Box-Ljung tests above failed to reject the null hypothesis for all three models to support that our residuals are independently distributed.

First and Seasonal Differenced Models (d = 1 , D = 1)

Next, we fit models that are both First Differenced and Seasonally Differenced and examine the residuals of these.

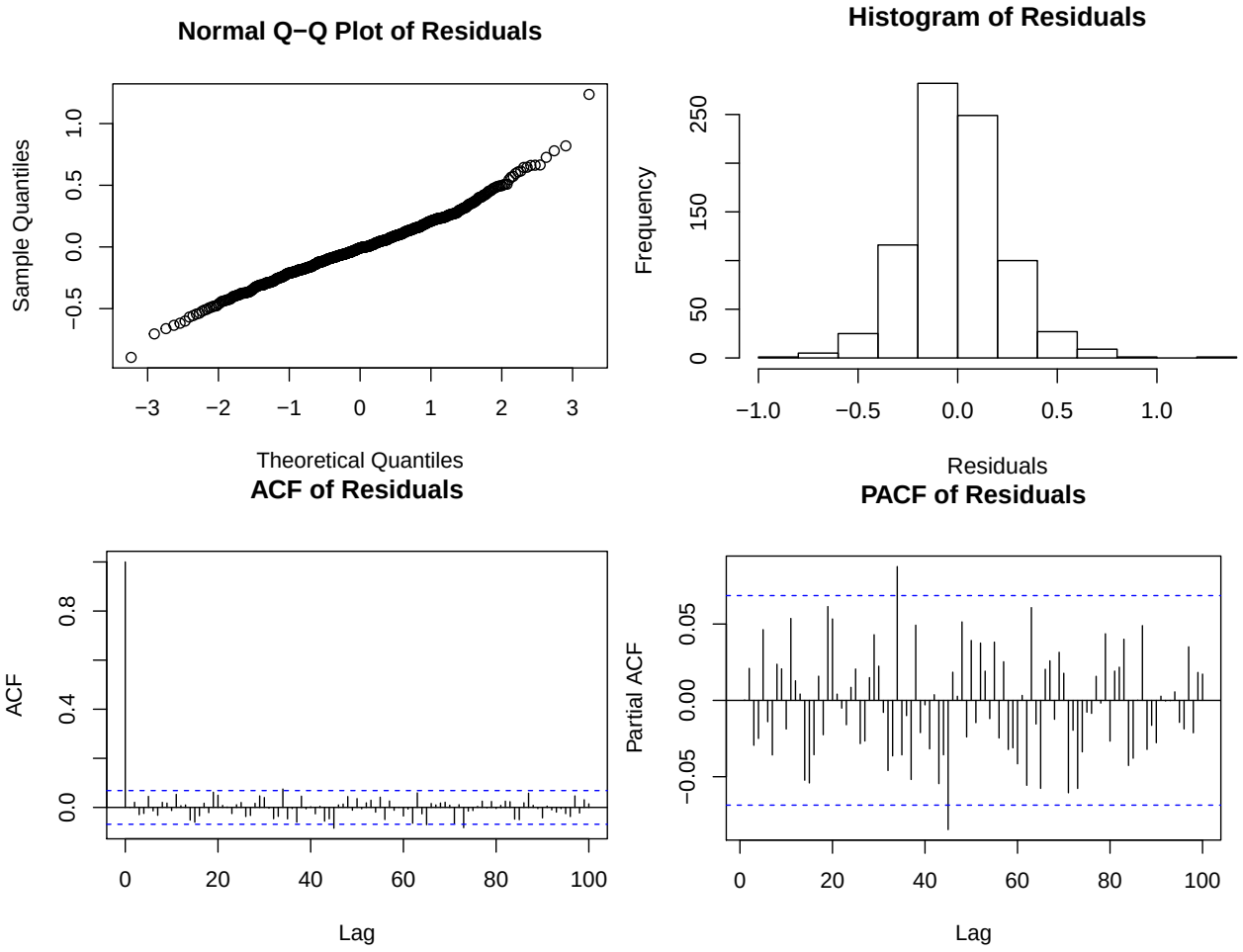
```
# top (2,1,3)(1,1,1)
m.diff.diff12.1 <- Arima(unem.train.ts, order = c(2, 1,
    3), seasonal = list(order = c(1, 1, 1), period = 12))

# 2nd (2,1,1)(0,1,1)
m.diff.diff12.2 <- Arima(unem.train.ts, order = c(2, 1,
    1), seasonal = list(order = c(0, 1, 1), period = 12))
```

```
# top (2,1,3)(1,1,1)
print_resid_chart(m.diff.diff12.1)
```

```
## Model SARIMA  2 1 3 1 1 1 :
## AIC:  -28.38205
## BIC:  9.124788
```

```
# top (2,1,3)(1,1,1)
Box.test(m.diff.diff12.1$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
## data:  m.diff.diff12.1$residuals
## X-squared = 20.59, df = 25, p-value = 0.7152
```



We begin with the model which has the lowest AIC within its search space, the SARIMA(2,1,3)(1,1,1)[12]. We note again that the residuals for this model are approximately normally distributed, with the exception of one outlier. Both the ACF and PACF plots show no more significant spikes than what would happen by chance. The residuals for this plot look very similar to white noise. The Ljung-Box test also fails to reject the null hypothesis that the residuals are independently distributed.

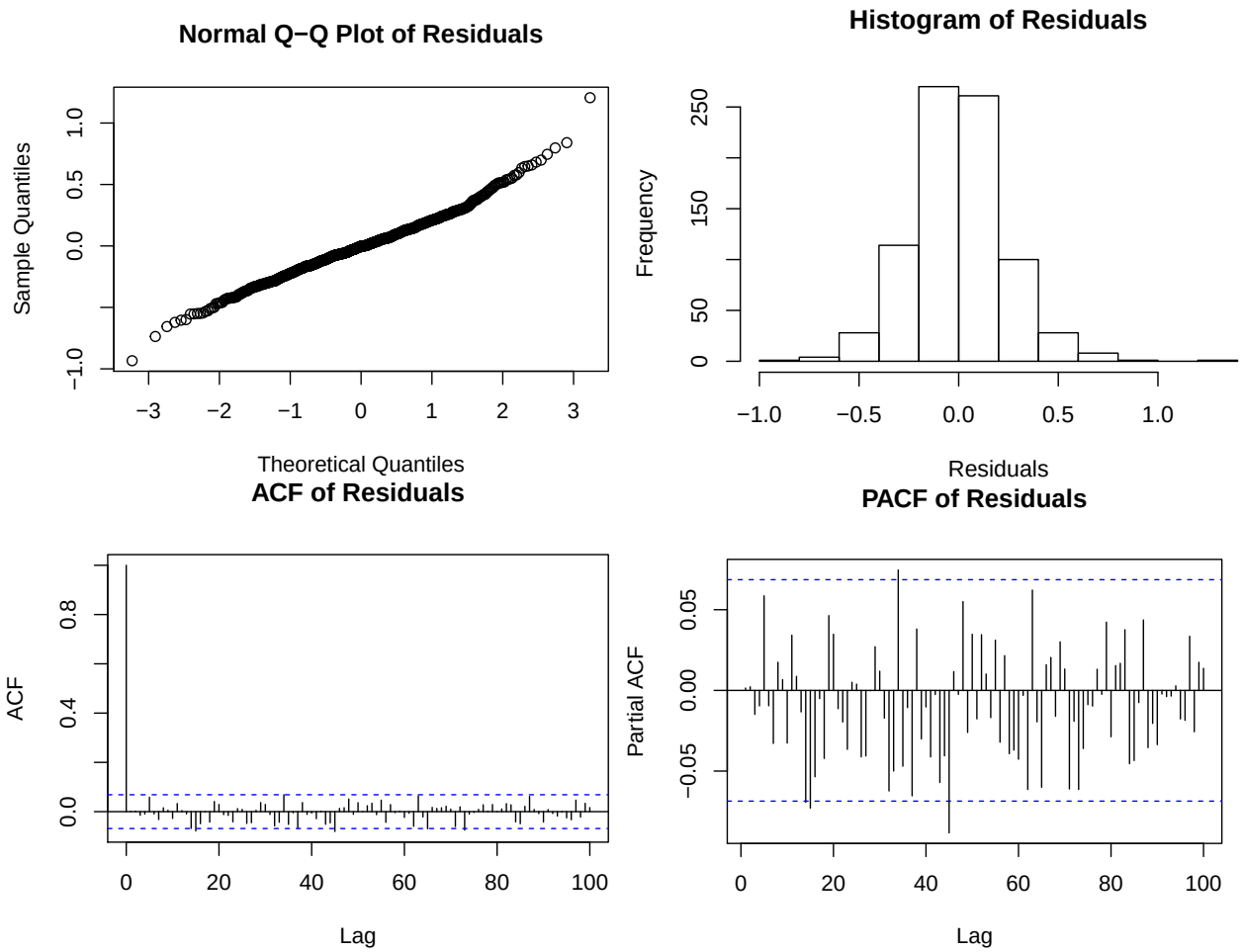
```
# 2nd (2,1,1)(0,1,1)
print_resid_chart(m.diff.diff12.2)
```

```
## Model SARIMA  2 1 1 0 1 1 :
## AIC:  -18.35035
## BIC:  5.091422
```

```
# 2nd (2,1,1)(0,1,1)
Box.test(m.diff.diff12.2$residuals, lag = 25, type = c("Ljung-Box"))
```

```
##
##  Box-Ljung test
##
```

```
## data:  m.diff.diff12.2$residuals
## X-squared = 22.384, df = 25, p-value = 0.6135
```



If we simplify this model to a SARIMA(2,1,1)(0,1,1)[12] the residual diagnostic plots look fairly similar. The PACF plot does show a few more significant spikes, but there are still fewer than what might happen by chance, so we are not concerned that we have lost our well-behaved residuals as a result of simplifying the model. The Ljung-Box test again fails to reject the null hypothesis that the residuals are independently distributed.

For first and seasonal differenced models, we reduced from the top AIC candidate, SARIMA(2,1,3)(1,1,1) to the 2nd candidate SARIMA(2,1,1)(0,1,1) with similar white noise behaviors. We will keep both of these candidates to compare out of sample performance:

- SARIMA(2,1,3)(1,1,1)
- SARIMA(2,1,1)(0,1,1)

Compare models based on out of sample errors until June 2017

(A.i)How well does your model predict the unemployment rate up until June 2017?

```
# candidate models SARIMA(4,0,3)(0,1,1) m.diff12.1
# SARIMA(2,0,1)(0,1,1) m.diff12.7 SARIMA(8,1,3)(1,0,1)
# m.diff.1 SARIMA(2,1,1)(1,0,1) m.diff.3
# SARIMA(1,1,1)(1,0,1) m.diff.6 SARIMA(2,1,3)(1,1,1)
# m.diff.diff12.1 SARIMA(2,1,1)(0,1,1) m.diff.diff12.2
```

```
candidate_mods = list(m.diff12.1, m.diff12.7, m.diff.1,
    m.diff.3, m.diff.6, m.diff.diff12.1, m.diff.diff12.2)
```

In order to measure the out of sample fit, we calculate the Root Mean Squared Error of each model's forecast compared to the observed data from January 2016-June 2017. The Root Mean Squared Error is given as:

$RMSE = \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$

where y_i is the observed data and $\hat{y}_i$ is the forecasted data.

```
# function to get RMSE
get_RMSE = function(test.df, mod, ahead) {
    f = forecast(mod, ahead)$mean
    sq.error = (test.df$UNRATENSA - f)^2
    rmse = sqrt(mean(sq.error))
    return(data.frame(p = mod$arma[1], d = mod$arma[6],
        q = mod$arma[2], P = mod$arma[3], D = mod$arma[7],
        Q = mod$arma[4], RMSE = rmse))
}
```

```
RMSE.df = data.frame()
for (i in 1:length(candidate_mods)) {
    add.df = get_RMSE(unem.test, candidate_mods[[i]], 18)
    RMSE.df = rbind(RMSE.df, add.df)
}
RMSE.df = RMSE.df[order(RMSE.df$RMSE), ]
RMSE.df
```

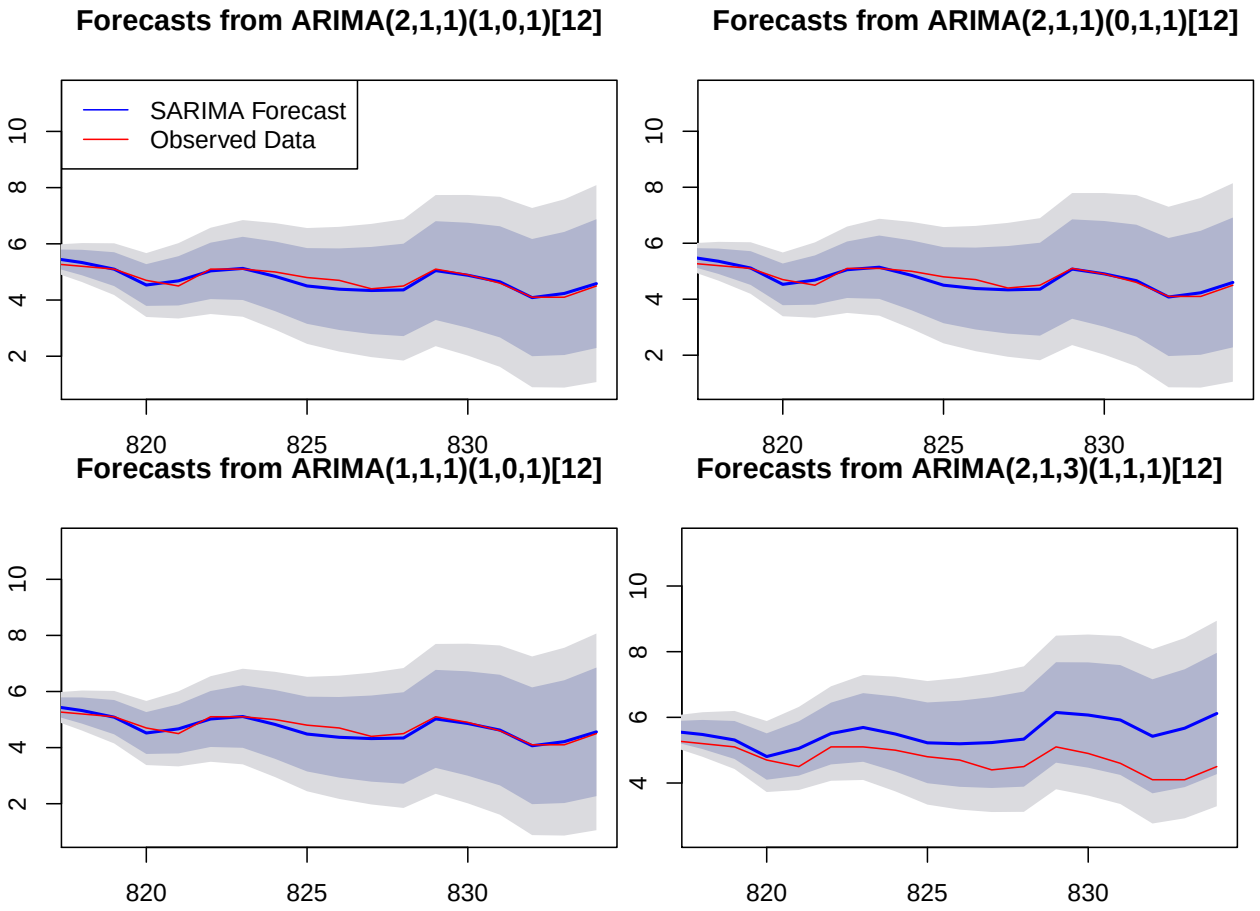
```
##   p d q P D Q      RMSE
## 4 2 1 1 1 0 1 0.1472144
## 7 2 1 1 0 1 1 0.1510539
## 5 1 1 1 1 0 1 0.1516677
## 3 8 1 3 1 0 1 0.8159746
## 6 2 1 3 1 1 1 0.8859527
## 2 2 0 1 0 1 1 0.8890228
## 1 4 0 3 0 1 1 0.8946334
```

Out of sample errors tells us that SARIMA(2,1,1)(1,0,1), SARIMA(2,1,1)(0,1,1), and SARIMA(1,1,1)(1,0,1) perform superior to the other candidates and their RMSEs are very close. We plot their forecasts to further compare these three. We also include a plot of the model with the residuals that most closely resembled white noise, a SARIMA(2,1,3)(1,1,1)[12.]

```
unem.actual.ts = ts(unem.data$UNRATENSA)
# (2,1,1)(1,0,1)
f.diff.3 <- forecast(m.diff.3, 18)
# (2,1,1)(0,1,1)
f.diff.diff12.2 <- forecast(m.diff.diff12.2, 18)
# (1,1,1)(1,0,1)
f.diff.6 <- forecast(m.diff.6, 18)
# (2,1,3)(1,1,1)
f.diff.diff12.1 <- forecast(m.diff.diff12.1, 18)
```

```
plot(f.diff.3, xlim = c(818, 834))
lines(unem.actual.ts, type = "l", col = "red")
legend("topleft", col = c("blue", "red"), legend = c("SARIMA Forecast",
    "Observed Data"), lty = c(1, 1))
```

```r
plot(f.diff.diff12.2, xlim = c(818, 834))
lines(unem.actual.ts, type = "l", col = "red")
plot(f.diff.6, xlim = c(818, 834))
lines(unem.actual.ts, type = "l", col = "red")
plot(f.diff.diff12.1, xlim = c(818, 834))
lines(unem.actual.ts, type = "l", col = "red")
```



All of the first three models predict very closely to the test data. SARIMA(2,1,1)(1,0,1) performs the closest in the time plots, and it has the lowest RMSE of 0.147.

Based on the criteria of the performance of residuals and the out of sample fit, we have two conflicting choices for the "best" model.

Based on the performance of the residuals, a SARIMA(2,1,3)(1,1,1)[12] had residuals that most closely resembled white noise. However, as seen from the plot above, it does a poor job of accurately predicting the test data.

Based on the out of sample fit, the SARIMA(2,1,1)(1,0,1)[12] performs the best, since it very accurately predicted the test data from January 2016-June 2017. We recall that the residuals for this model were not considerably different from white noise.

Since our primary task is to forecast unemployment, we will prioritize the out of sample fit and choose the SARIMA(2,1,1)(1,0,1)[12] as our best model.

We will use this model to forecast unemployment rate until 2020. Before that, the model specification is given by the following:

```
m.diff.3$coef
```

```
##        ar1        ar2        ma1       sar1       sma1
##  0.5985857  0.1241366 -0.4845903  0.9918375 -0.7582691
```

```
# characteristic equation for differenced AR component
Mod(polyroot(c(1, -0.5986, -0.1241)))
```

```
## [1] 1.313101 6.136631
```

```
# characteristic equation for differenced SAR component
Mod(polyroot(c(1, -0.9918)))
```

```
## [1] 1.008268
```

```
# characteristic equation for differenced MA component
Mod(polyroot(c(1, -0.4846)))
```

```
## [1] 2.063558
```

```
# characteristic equation for differenced SMA component
Mod(polyroot(c(1, -0.7583)))
```

```
## [1] 1.318739
```

$$(1-\Theta B^{12})(1-\theta_1 B-\theta_2 B^2)(1-B)x_t=(1+\Phi B^{12})(1+\phi B)w_t$$

where $\Theta=+0.9918$, $\theta_1=+0.5986$, $\theta_2=+0.1241$, $\Phi=-0.7583$, $\phi=-0.4846$.

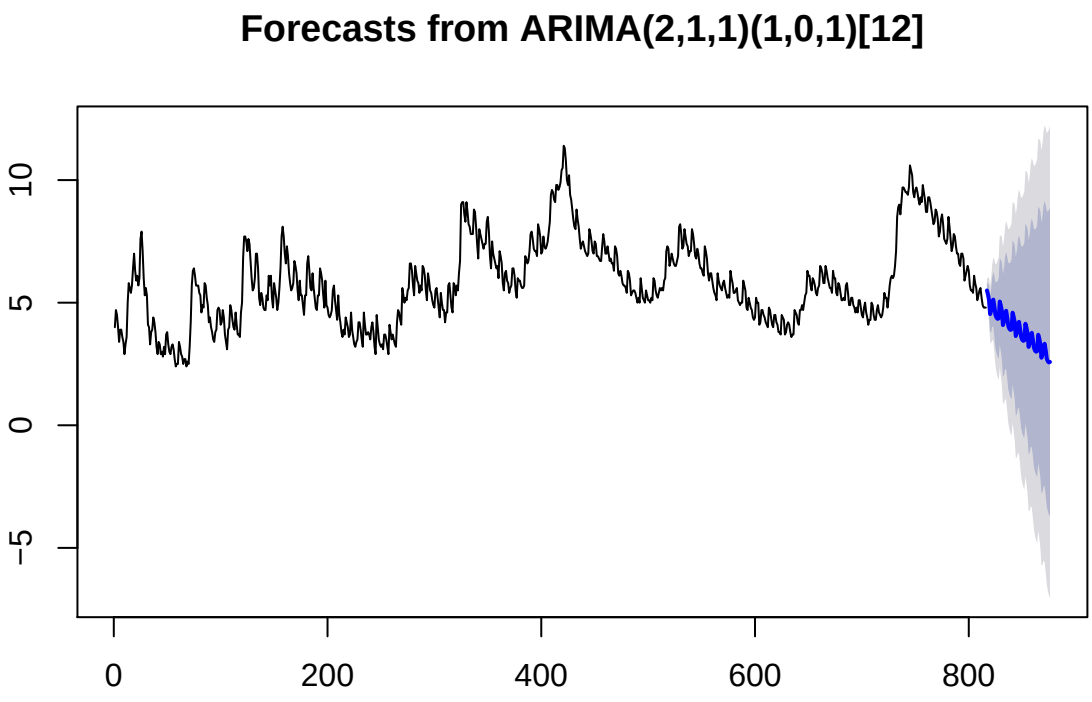
Based on this specification, we note the following:

- A unit root on the left specified by $(1-B)$, which was taken care of by first differencing.
- The first differenced AR component has characteristic equation $1-0.5986B-0.1241B^2=0$, the roots for B are 1.313 and 6.136. The first differenced seasonal component has characteristic equation $1-0.9918B^{12}=0$, the root for B^{12} is 1.008. All roots exceed unity which means the first differenced series is stationary.
- The first differenced MA component has characteristic equation $1-0.4846B=0$, the roots for B is 2.064. The first differenced seasonal MA component has characteristic equation $1-0.7583B^{12}=0$, the roots for B^{12} is 1.319. All roots exceed unity which means the first differenced series is invertible.

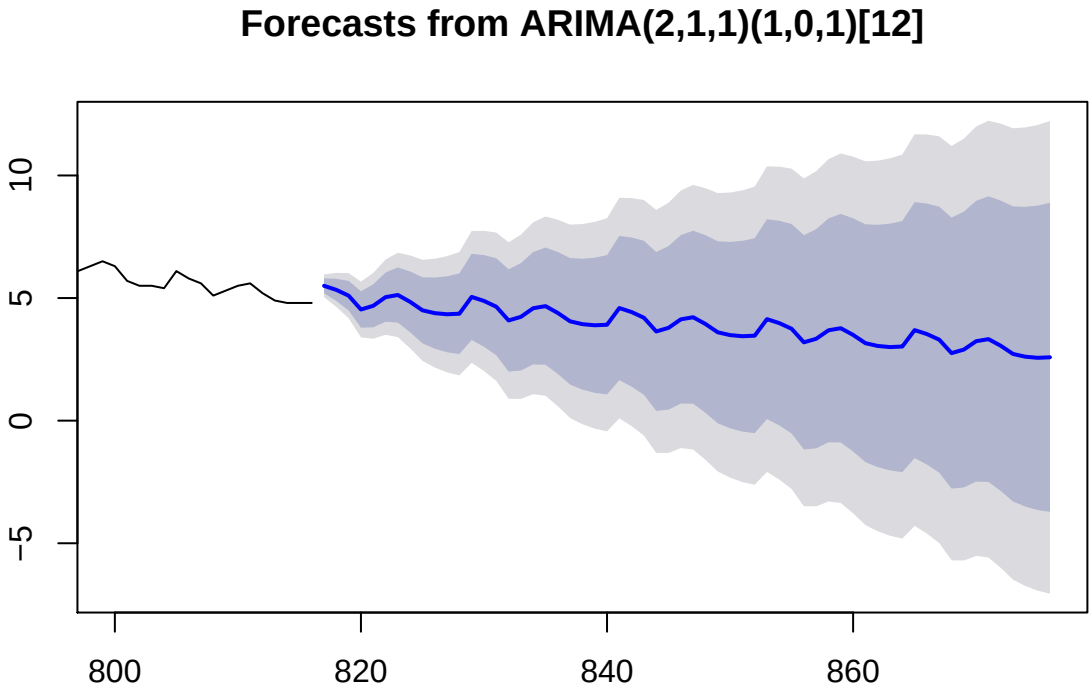
Forecast until 2020

(A.ii) What does the unemployment rate look like at the end of 2020? How credible is this estimate?

```
# 817 2016 Jan forecast.sarima.ts =
# ts(forecast(m.diff.3,h=60)$mean, start = c(2016,1),
# frequency = 12)

par(mfrow = c(1, 1))

# forecast out to Dec 2020
plot(forecast(m.diff.3, h = 60))
```




```
plot(forecast(m.diff.3, h = 60), xlim = c(800, 876))
```




```
f.diff.3.2020 <- forecast(m.diff.3, h = 60)

expected <- f.diff.3.2020$mean[60]
lowerbound <- f.diff.3.2020$lower[60, 2]
upperbound <- f.diff.3.2020$upper[60, 2]
```

Forecast 2020 December – Expected Value: 2.5835788

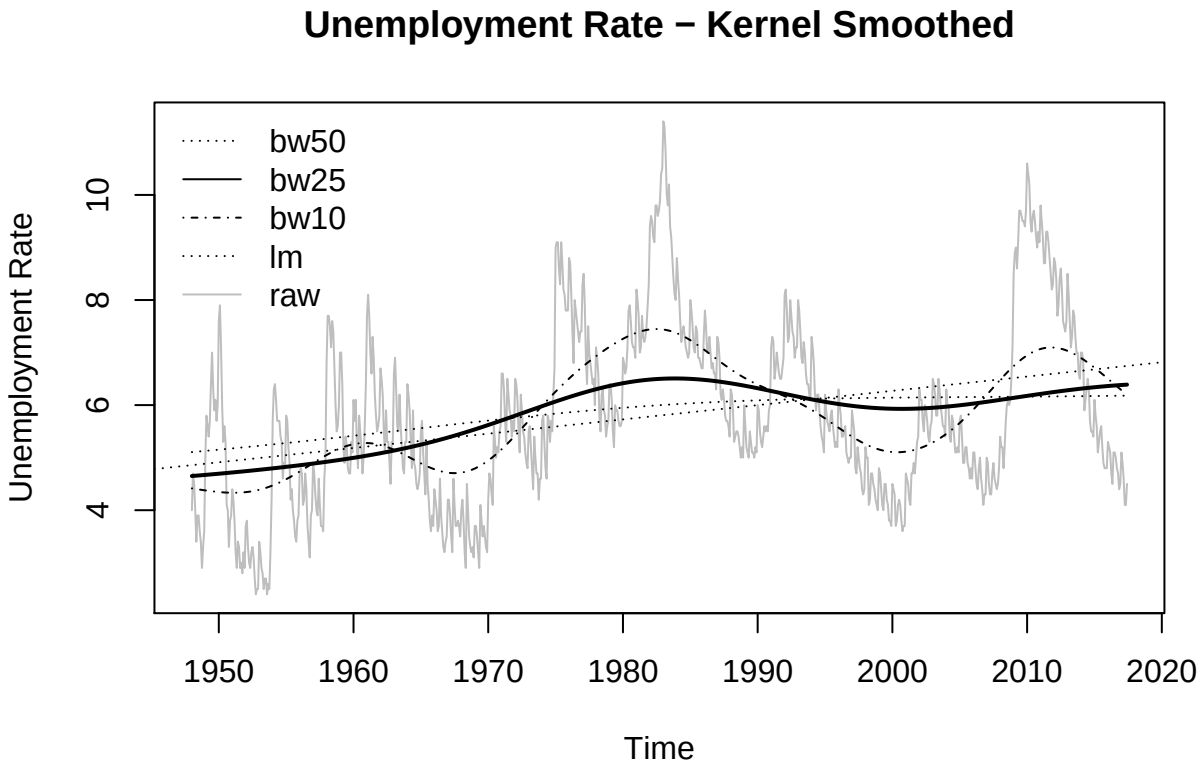
Forecast 2020 December – 95% Lower Confidence Bound: -7.0546782

Forecast 2020 December – 95% Upper Confidence Bound: 12.2218357

As we can see from the forecast time plots above, the confidence bounds of forecast expand drastically towards 2020. Although the mean value is expected to be 2.5835, this estimate is not very credible. The confidence interval is so wide that either a continued decreasing or increasing trend is possible. One additional concern is that the confidence interval includes negative values, which is not possible for an unemployment rate.

(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.

```
# Kernel smoothing
unem.k.smooth.widest = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 50)

unem.k.smooth.wide = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 25)

unem.k.smooth.narrow = ksmooth(time(unem.ts), unem.ts, kernel = c("normal"),
    bandwidth = 10)
# Make plot
plot(unem.ts, col = "gray", ylab = "Unemployment Rate",
    main = "Unemployment Rate - Kernel Smoothed")
lines(unem.k.smooth.widest$x, unem.k.smooth.widest$y, col = "black",
    lty = "dotted")
lines(unem.k.smooth.wide$x, unem.k.smooth.wide$y, col = "black",
    lty = "solid", lwd = 2)
lines(unem.k.smooth.narrow$x, unem.k.smooth.narrow$y, col = "black",
    lty = "dotdash")
abline(lm(unem.ts ~ time(unem.ts)), lty = "dotted", col = "black")
legend("topleft", legend = c("bw50", "bw25", "bw10", "lm",
    "raw"), lty = c("dotted", "solid", "dotdash", "dotted",
    "solid"), bty = "n", col = c("black", "black", "black",
    "black", "gray"))
```



Continuing the insights from our EDA, the time series of unemployment rate has an upward trend and shows some quadratic behavior. Therefore we can estimate a model with quadratic term of time. We attempt to explain seasonal behavior using a dummy variable for each month. Candidate Models are:

$$X_t = \beta_0 + \beta_t time + \beta_m month + \epsilon$$

$$X_t = \beta_0 + \beta_t time + \beta_{t2} time^2 + \beta_{\mathbf{m}} \mathbf{month} + \epsilon$$

where *time* is an annual unit, each additional month is expressed as a fraction of the year. *month* is a categorical variable that is broken down into indicator variables.

```
# Preparing data
unem.ts = ts(unem.data$UNRATENSA, frequency = 12, start = c(1948,
    1))

unem.train.ts = window(unem.ts, end = c(2015, 12))
unem.test.ts = window(unem.ts, start = c(2016, 1))

y.lm = as.numeric(unem.train.ts)
t.lm = as.numeric(time(unem.train.ts))
mon.lm = as.factor(cycle(unem.train.ts))
```

```
# Fit Linear Models
unem.lm = lm(y.lm ~ t.lm + mon.lm)
unem.lm.quad = lm(y.lm ~ t.lm + I(t.lm^2) + mon.lm)

# Calculate AIC
lm.aic <- round(AIC(unem.lm), 2)
lm.quad.aic <- round(AIC(unem.lm.quad), 2)
```

```
# Calcualte Heteroskedastic Robust Standard Errors
se.lm = sqrt(diag(vcovHC(unem.lm)))
se.lm.quad = sqrt(diag(vcovHC(unem.lm.quad)))
```

```
stargazer(unem.lm, unem.lm.quad, se = list(se.lm, se.lm.quad),
    keep.stat = c("rsq", "adj.rsq", "n", "f"), add.lines = list(c("AIC",
        lm.aic, lm.quad.aic)), type = "latex")
```

% Table created by stargazer v.5.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Sun, Aug 06, 2017 - 21:56:35

Our quadratic model performs slightly better than the linear model by AIC and Adjusted R-squared (See Table 1). The F-statistic and F-test p-value strongly reject that null hypothesis that our coefficients are not jointly significant thus supporting explanatory power of our model.

```
car::linearHypothesis(unem.lm.quad, c("mon.lm2 = 0", "mon.lm3 = 0",
    "mon.lm4 = 0", "mon.lm5 = 0", "mon.lm6 = 0", "mon.lm7 = 0",
    "mon.lm8 = 0", "mon.lm9 = 0", "mon.lm10 = 0", "mon.lm11 = 0",
    "mon.lm12 = 0"), vcov = vcovHC)
```

```
## Linear hypothesis test
##
## Hypothesis:
## mon.lm2 = 0
## mon.lm3 = 0
## mon.lm4 = 0
## mon.lm5 = 0
## mon.lm6 = 0
## mon.lm7 = 0
## mon.lm8 = 0
## mon.lm9 = 0
## mon.lm10 = 0
## mon.lm11 = 0
## mon.lm12 = 0
##
## Model 1: restricted model
## Model 2: y.lm ~ t.lm + I(t.lm^2) + mon.lm
##
## Note: Coefficient covariance matrix supplied.
##
##   Res.Df Df      F    Pr(>F)
## 1    813
## 2    802 11 4.6588 6.382e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The second F-test result also provides evidence that our month variable is significant. We will proceed with this quadratic model for a better fit. Our estimated model is specified as:

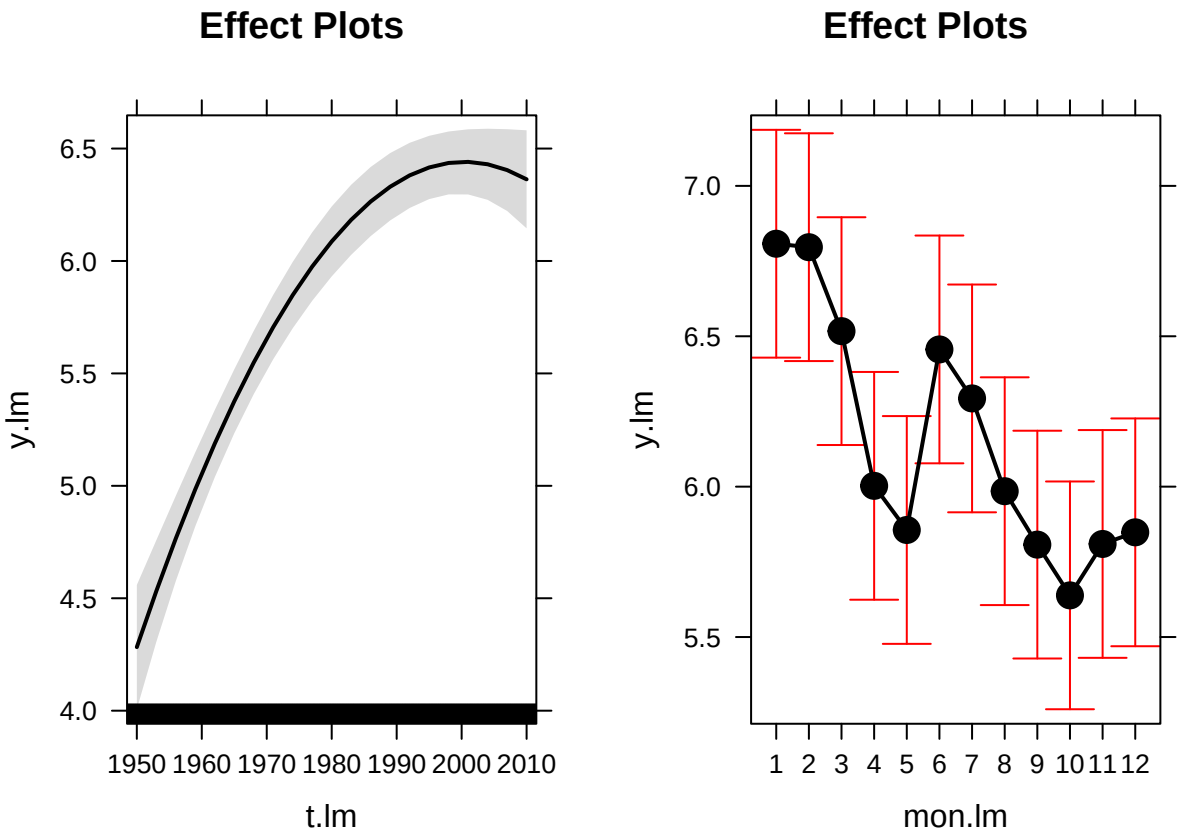
$\hat{X}_t = -3389 + 3.396time - 0.0008487time^2 + -0.805 \cdot I(Apr) - 0.9518 \cdot I(May) - 0.5144 \cdot I(Jul)$
$- 0.8229 \cdot I(Aug) - 1 \cdot I(Sep) - 1.169 \cdot I(Oct) - 0.9982 \cdot I(Nov) - 0.9596 \cdot I(Dec)$

Notice that we have dropped some month indicator variables in the specification, because they are not statistically different from the base variable January. Along with the negative sign of the significant indicator variables, the estimations agrees with our earlier month plot in the EDA section that the beginning and middle of each year tend to have higher unemployment rates.

Table 1:

	Dependent variable:	
	y.lm	
	(1)	(2)
t.lm	0.031^{***}	3.396^{***}
	(0.003)	(0.629)
I(t.lmˆ2)		-0.001^{***}
		(0.0002)
mon.lm2	-0.011	-0.011
	(0.282)	(0.278)
mon.lm3	-0.291	-0.291
	(0.280)	(0.276)
mon.lm4	-0.805^{***}	-0.805^{***}
	(0.274)	(0.270)
mon.lm5	-0.952^{***}	-0.952^{***}
	(0.273)	(0.269)
mon.lm6	-0.351	-0.351
	(0.273)	(0.268)
mon.lm7	-0.514^{*}	-0.514^{*}
	(0.271)	(0.267)
mon.lm8	-0.823^{***}	-0.823^{***}
	(0.269)	(0.265)
mon.lm9	-1.000^{***}	-1.000^{***}
	(0.268)	(0.262)
mon.lm10	-1.169^{***}	-1.169^{***}
	(0.269)	(0.263)
mon.lm11	-0.998^{***}	-0.998^{***}
	(0.267)	(0.261)
mon.lm12	-0.960^{***}	-0.960^{***}
	(0.270)	(0.266)
Constant	-55.652^{***}	$-3{,}389.298^{***}$
	(5.375)	(623.107)
AIC	3036.81	3008.38
Observations	816	816
R^2	0.183	0.213
Adjusted R^2	0.171	0.200
F Statistic	15.016^{***} (df = 12; 803)	16.713^{***} (df = 13; 802)
Note:		*p<0.1; **p<0.05; ***p<0.01

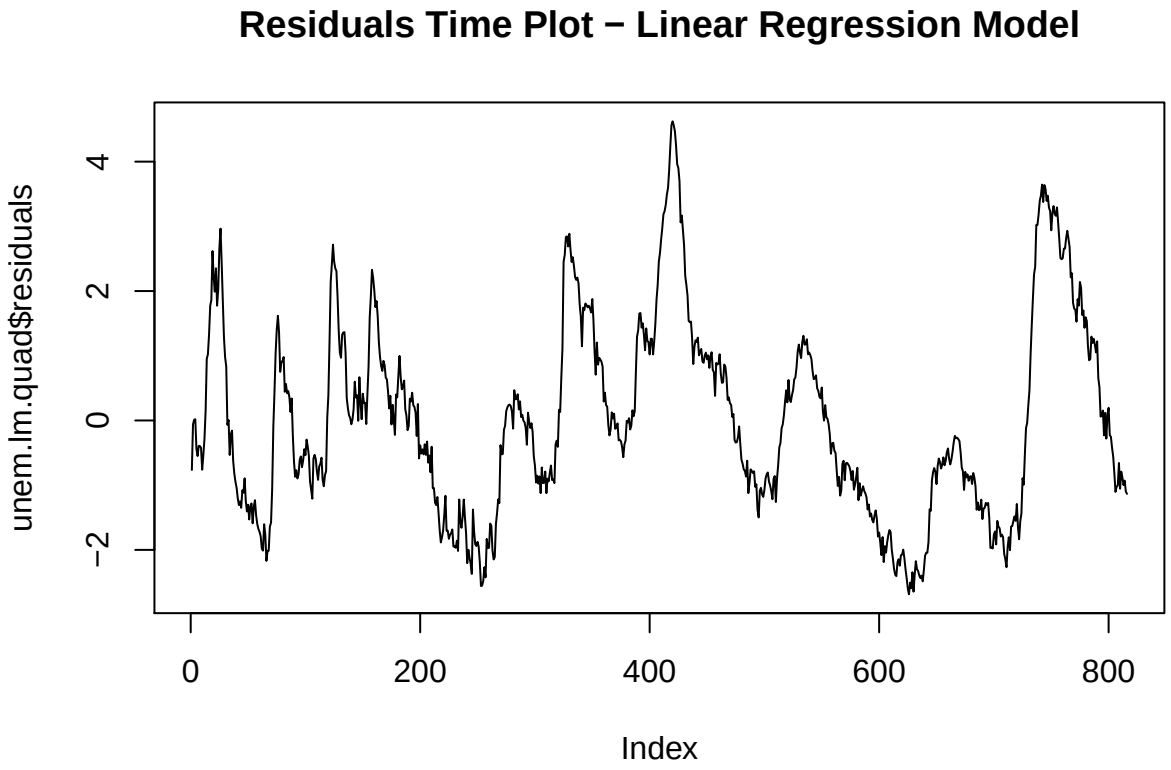
```
plot(effects::allEffects(unem.lm.quad)[c(1, 3)], main = "Effect Plots")
```



The time effect plot above shows the fitted curve with a positive, gradually leveling slope and expanding confidence interval with the increase in time. Also, the month effect plot shows discrete levels and the model clearly distinguishes early and mid year from the other months. Both plots align with our EDA findings. To further examine the internal validity of our model, we evaluate the validity of our model by the 6 CLM assumptions:

1. Linearity in Parameters: This is a weak assumption, we have specified our model with linear coefficients.
2. Random sample of data: We clearly have violated this assumption because the observations are serially correlated, as demonstrated in the autocorrelation plots in the EDA. The strongly time dependent residuals plotted below demonstrates that our observations could not have been independent in the first place.

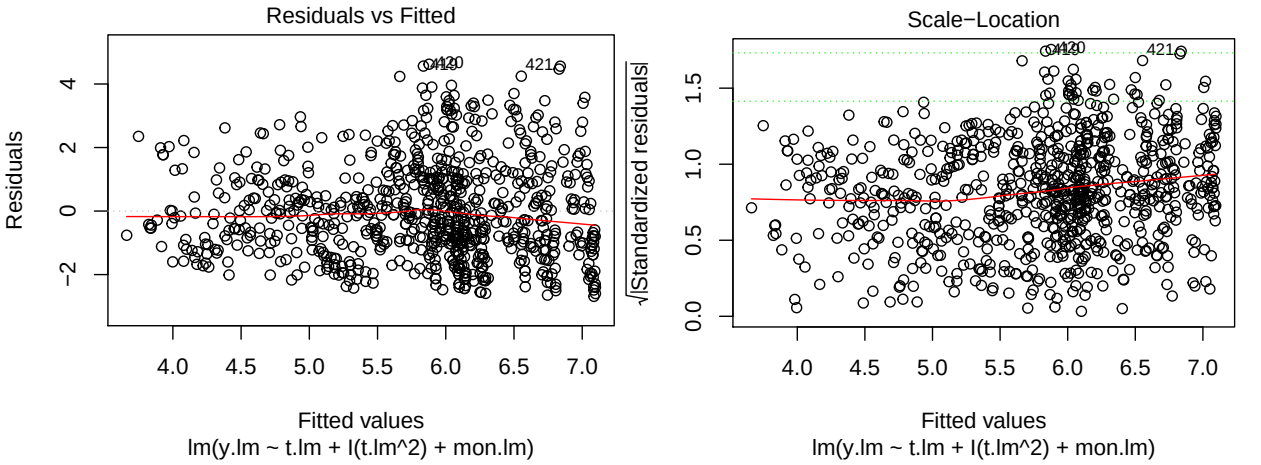
```
plot(unem.lm.quad$residuals, type = "l", main = "Residuals Time Plot - Linear Regression Model")
```



3. No perfect co-linearity: Each year has 12 months, therefore the time variable is not correlated with month.

4. Zero-conditional mean. From the residuals vs fitted values plot, there is a clear curvature of the loess curve from line zero. We tried a separate model with the cubic term of time but the curvature was only flipped not flattened. We could be missing an important variable here. This assumption is violated.

```
plot(unem.lm.quad, which = 1)
plot(unem.lm.quad, which = 3)
abline(h = c(sqrt(2), sqrt(3)), col = "green", lty = "dotted")
```



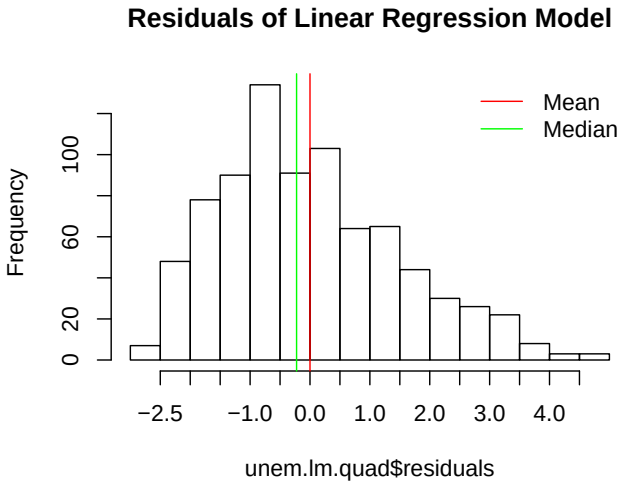
5. Homoskedasticity of errors: The variance of residuals noticeably expands towards higher fitted values. The Loess curve on the scale-location plot clearly picks up at the same time. The Breusch-Pagan test strongly rejects the null hypothesis of homoskedasticity. This assumption is violated.

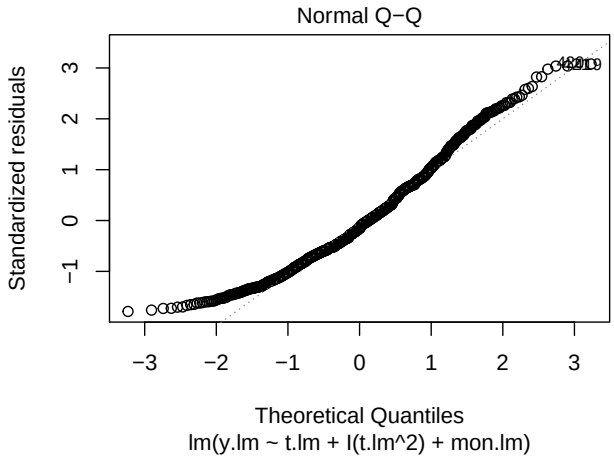
```
lmtest::bptest(unem.lm.quad)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  unem.lm.quad
## BP = 29.662, df = 13, p-value = 0.005268
```

6. Normally distributed error: From the Normal Q-Q plot and histogram, our residuals are clearly right skewed. The Shapiro test also strongly rejects the null hypothesis that our residuals are normally distributed.

```
hist(unem.lm.quad$residuals, xaxt = "n", main = "Residuals of Linear Regression Model")
axis(side = 1, at = seq(-2.5, 4.5, 0.5))
abline(v = mean(unem.lm.quad$residuals), col = "red")
abline(v = median(unem.lm.quad$residuals), col = "green")
legend("topright", legend = c("Mean", "Median"), col = c("red",
    "green"), bty = "n", lty = c("solid", "solid"))

plot(unem.lm.quad, which = 2)
```



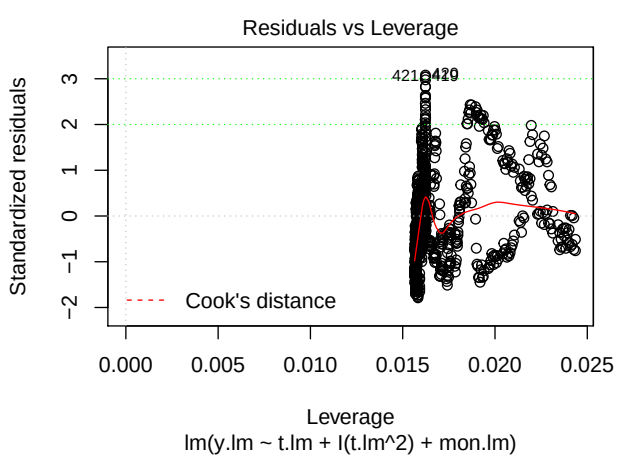


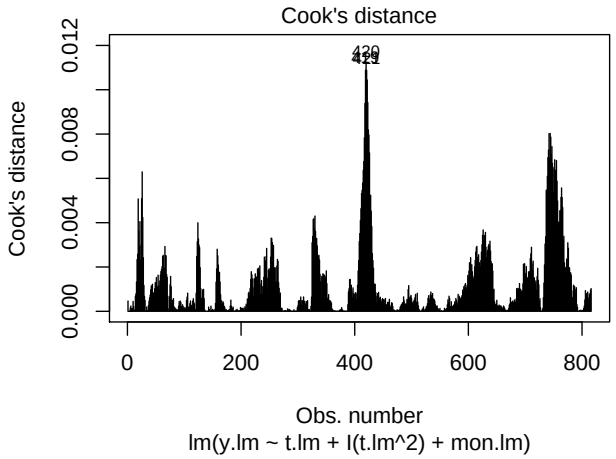

```
shapiro.test(unem.lm.quad$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  unem.lm.quad$residuals
## W = 0.96731, p-value = 1.513e-12
```

7. Outlier Analysis (Not a CLM assumption): From the scale-location plot above, we see a number of standardized residuals lying outside the threshold of 2 and 3 standard deviations. This says that our model represents variations in our data poorly. None of the observations are close to cook's distance of 0.5 so there are no extreme outliers.

```
plot(unem.lm.quad, which = 5)
abline(h = c(2, 3), col = "green", lty = "dotted")
plot(unem.lm.quad, which = 4)
```





(B.i) How well does your model predict the unemployment rate up until June 2017? (B.ii) What does the unemployment rate look like at the end of 2020? How credible is this estimate? (B.iii) Compare this forecast to the one produced by the SARIMA model. What do you notice?

Prediction Through 2017

```
test.time = seq(2016, 2017.417, by = (1/12))
test.month = c(seq(1, 12), seq(1, 6))

test.prediction = predict.lm(object = unem.lm.quad, se.fit = T,
    newdata = data.frame(t.lm = test.time, mon.lm = as.factor(test.month)))
```

```
data.frame(year = floor(test.time), month = test.month,
    `lm mean` = test.prediction$fit, `lm error` = test.prediction$fit -
        unem.test$UNRATENSA, `SARIMA mean` = as.numeric(f.diff.3$mean),
    `SARIMA error` = as.numeric(f.diff.3$mean) - unem.test$UNRATENSA)
```

```
##    year month  lm.mean lm.error SARIMA.mean SARIMA.error
## 1  2016     1 6.891331 1.591331    5.498936  0.198936381
## 2  2016     2 6.877627 1.677627    5.333462  0.133461832
## 3  2016     3 6.596276 1.496276    5.098298 -0.001701837
## 4  2016     4 6.079631 1.379631    4.533827 -0.166172602
## 5  2016     5 5.930634 1.430634    4.683256  0.183255574
## 6  2016     6 6.528695 1.428695    5.037484 -0.062515551
## 7  2016     7 6.363520 1.263520    5.125235  0.025234937
## 8  2016     8 6.052758 1.052758    4.844588 -0.155411579
## 9  2016     9 5.872877 1.072877    4.500414 -0.299585809
## 10 2016    10 5.701821 1.001821    4.384869 -0.315130537
## 11 2016    11 5.870470 1.470470    4.338786 -0.061213654
## 12 2016    12 5.906766 1.406766    4.360144 -0.139855527
## 13 2017     1 6.864047 1.764047    5.046995 -0.053004672
## 14 2017     2 6.850201 1.950201    4.881704 -0.018296485
## 15 2017     3 6.568709 1.968709    4.646968  0.046968193
## 16 2017     4 6.051923 1.951923    4.086067 -0.013932679
## 17 2017     5 5.902783 1.802783    4.233470  0.133469610
## 18 2017     6 6.500703 2.000703    4.584196  0.084195734
```

```
sarima.err = as.numeric(f.diff.3$mean) - unem.test$UNRATENSA
lm.err = test.prediction$fit - unem.test$UNRATENSA
```

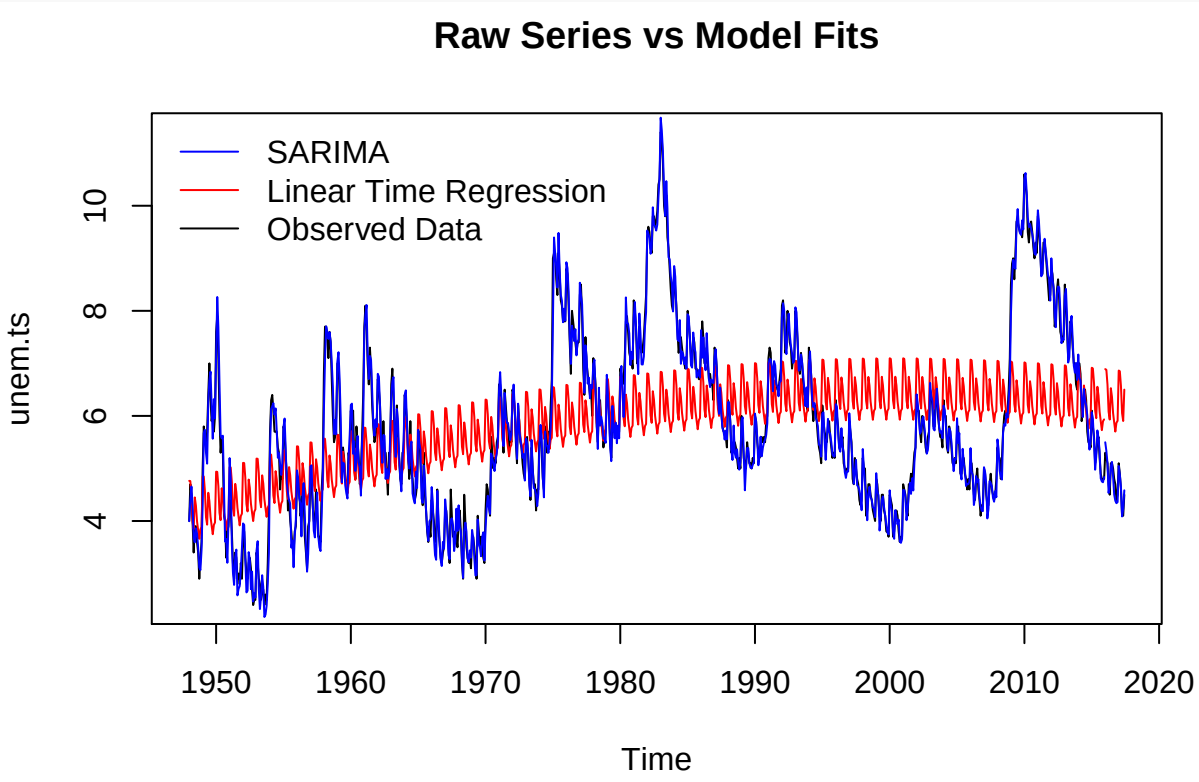
```
sarima.rmse = sqrt(mean(sarima.err^2))
lm.rmse = sqrt(mean(lm.err^2))
```

SARIMA RMSE: 0.1472144

Linear Model RMSE: 1.5711185

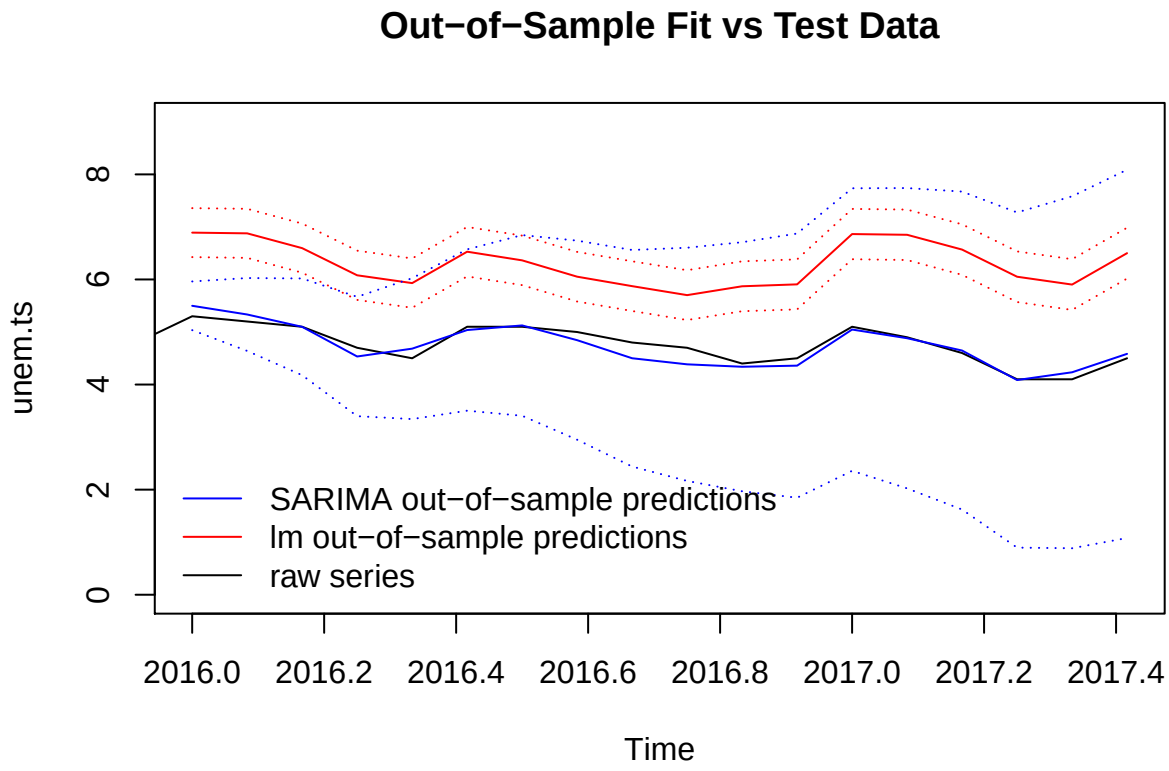
In comparison with the ARIMA model, the linear regression model has much poorer out-of-sample performance. The above table shows that the linear model predictions are consistently off by 1 to 2 units, while the SARIMA model predictions are only off from -0.4 to +0.2 units. Also, the RMSE of our SARIMA model is less than 10% of the RMSE of the linear model.

```
plot(unem.ts, xlim = c(1948, 2017.417), main = "Raw Series vs Model Fits")

lines(y = unem.lm.quad$fit, x = t.lm, col = "red")
lines(y = test.prediction$fit, x = test.time, col = "red")

lines(y = m.diff.3$fitted, x = t.lm, col = "blue")
lines(y = as.numeric(f.diff.3$mean), x = test.time, col = "blue")

legend("topleft", col = c("blue", "red", "black"), legend = c("SARIMA",
    "Linear Time Regression", "Observed Data"), bty = "n",
    lty = c(1, 1, 1))
```



The above long-term time plot shows that the fitted curve of our linear model fails to pick up most of the random walk variations in the observed series, while the step by step fit of the SARIMA model is quite close. Unless we fit a very high order term for time, it's impossible for the linear model to pick up such variations. On the other hand, seasonal patterns are reasonably approximated by both models; we can see the small ripples in the raw series replicated and matched by both fitted curves.

```r
plot(unem.ts, xlim = c(2016, 2017.417), ylim = c(0, 9),
    main = "Out-of-Sample Fit vs Test Data")

lines(y = test.prediction$fit, x = test.time, col = "red")
lines(y = test.prediction$fit + 1.96 * test.prediction$se.fit,
    x = test.time, col = "red", lty = "dotted")
lines(y = test.prediction$fit - 1.96 * test.prediction$se.fit,
    x = test.time, col = "red", lty = "dotted")

lines(y = f.diff.3$mean, x = test.time, col = "blue")
lines(y = as.numeric(f.diff.3$lower[, 2]), x = test.time,
    col = "blue", lty = "dotted")
lines(y = as.numeric(f.diff.3$upper[, 2]), x = test.time,
    col = "blue", lty = "dotted")

legend("bottomleft", col = c("blue", "red", "black"), legend = c("SARIMA out-of-sample predictions",
    "lm out-of-sample predictions", "raw series"), bty = "n",
    lty = c("solid", "solid", "solid"))
```



Within our test region, we see a clear trade-off between the two models. On the one hand, the mean estimates of the SARIMA model are quite close to the test data, but its confidence bound widens drastically towards mid-2017 so our predictions lack precision. On the other hand for the linear model, even the lower bound of the estimates are consistently far above the test data. However, the confidence bound is quite narrow. Therefore, the SARIMA model makes less biased but less precise estimates and the linear model makes more biased but more precise estimates.

Forecast Through 2020

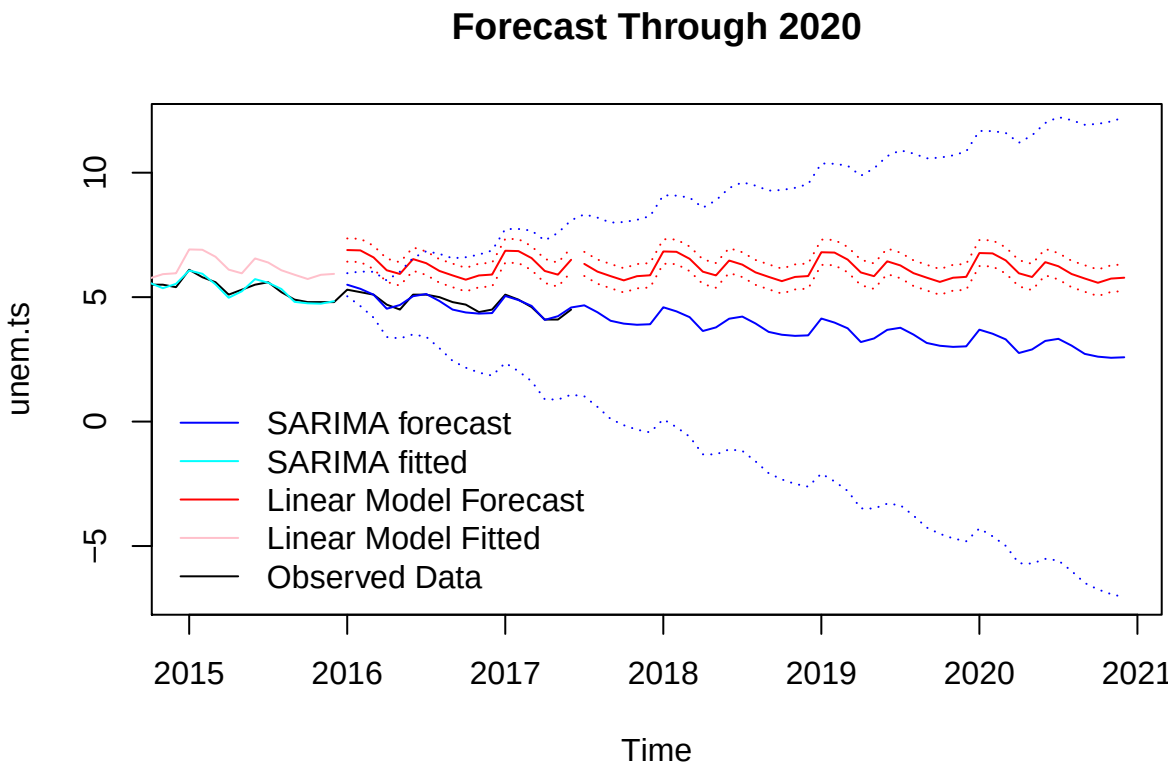
```
new.time = seq(2017.5, 2020.917, by = (1/12))
new.month = c(seq(7, 12), seq(1, 12), seq(1, 12), seq(1,
    12))

new.prediction = predict.lm(object = unem.lm.quad, se.fit = T,
    newdata = data.frame(t.lm = new.time, mon.lm = as.factor(new.month)))


new.df = data.frame(year = floor(new.time), month = new.month,
    `lm mean estimate` = new.prediction$fit, `lm upper estimate` = new.prediction$fit +
        1.96 * new.prediction$se.fit, `lm lower estimate` = new.prediction$fit -
        1.96 * new.prediction$se.fit)

tail(new.df, 12)
```

```
##    year month lm.mean.estimate lm.upper.estimate lm.lower.estimate
## 31 2020     1         6.772009          7.294879          6.249138
## 32 2020     2         6.757739          7.281795          6.233683
## 33 2020     3         6.475822          7.001068          5.950576
## 34 2020     4         5.958612          6.485052          5.432171
## 35 2020     5         5.809048          6.336688          5.281408
## 36 2020     6         6.406543          6.935386          5.877700
## 37 2020     7         6.240803          6.770854          5.710752
## 38 2020     8         5.929474          6.460737          5.398211
## 39 2020     9         5.749028          6.281507          5.216549
## 40 2020    10         5.577406          6.111105          5.043706
## 41 2020    11         5.745489          6.280414          5.210565
## 42 2020    12         5.781220          6.317373          5.245066
```

The above table shows that forecast of the linear model in 2020 oscillates within 5.5 percent and 6.8 percent. Its confidence intervals don't seem to fluctuate much either.

```
plot(unem.ts, xlim = c(2015, 2020.917), ylim = c(-7, 12),
    main = "Forecast Through 2020")

lines(y = unem.lm.quad$fit, x = t.lm, col = "pink")

lines(y = test.prediction$fit, x = test.time, col = "red")
lines(y = test.prediction$fit + 1.96 * test.prediction$se.fit,
    x = test.time, col = "red", lty = "dotted")
lines(y = test.prediction$fit - 1.96 * test.prediction$se.fit,
    x = test.time, col = "red", lty = "dotted")

lines(y = new.prediction$fit, x = new.time, col = "red")
lines(y = new.prediction$fit + 1.96 * new.prediction$se.fit,
    x = new.time, col = "red", lty = "dotted")
lines(y = new.prediction$fit - 1.96 * new.prediction$se.fit,
    x = new.time, col = "red", lty = "dotted")

lines(y = m.diff.3$fitted, x = t.lm, col = "cyan")

lines(y = f.diff.3.2020$mean, x = c(test.time, new.time),
    col = "blue")
```

```
lines(y = as.numeric(f.diff.3.2020$lower[, 2]), x = c(test.time,
    new.time), col = "blue", lty = "dotted")
lines(y = as.numeric(f.diff.3.2020$upper[, 2]), x = c(test.time,
    new.time), col = "blue", lty = "dotted")

legend("bottomleft", col = c("blue", "cyan", "red", "pink",
    "black"), legend = c("SARIMA forecast", "SARIMA fitted",
    "Linear Model Forecast", "Linear Model Fitted", "Observed Data"),
    bty = "n", lty = c("solid", "solid", "solid", "solid"))
```



The time plot above shows that the linear model's mean forecast oscillates around 6 percent without clear upward or downward overall trend, as does its confidence bound. On the other hand the SARIMA model's mean forecast continues the slight downward trend of the raw series and its confidence bound expands drastically to even below zero in 2020, which is nonsensical. For these reasons, we deem both models inadequate to forecast the unemployment rate as far as the end of 2020.

Question 3: VAR

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?

Training VAR Models

The EDA and plots above showed that both time series have trend and seasonality. Since a VAR model requires both time series to be weakly stationary, we need to remove the trend and seasonality when applying the VAR model. As for the VAR() function, it can incorporate trends and seasonality in the model. According to the documentation, the argument "type = trend" in this function only refers to a linear trend. In our case, it may be inappropriate to use the linear trend. Hence, we will explore taking the 1st difference of both time-series, then finding the best order by using VARselect(..., season = 12).

Continuing insights from the EDA, we attempt to train a model for each of the following differencing options:

- Both series (1) take first difference and (2) apply seasonal dummy variables
- Both series (1) take seasonal difference and (2) apply trend as regressor
- Both series (1) take first and seasonal difference
- Both series (1) take no difference (2) apply trend as regressor and (3) apply seasonal dummy variables

```
combined.raw = ts.intersect(unem.ts, auto.ts)

combined.raw.train = window(combined.raw, end = c(2015,
    12))
combined.raw.test = window(combined.raw, start = c(2016,
    1))

unem.var.train = combined.raw.train[, 1]
auto.var.train = combined.raw.train[, 2]
unem.var.test = combined.raw.test[, 1]
auto.var.test = combined.raw.test[, 2]
```

```
# 1st differenced
unem.var.train.diff = diff(unem.var.train)
auto.var.train.diff = diff(auto.var.train)

# 12th differenced
unem.var.train.diff12 = diff(unem.var.train, lag = 12)
auto.var.train.diff12 = diff(auto.var.train, lag = 12)

# 1st & 12th differenced
unem.var.train.diff.diff12 = diff(diff(unem.var.train),
    lag = 12)
auto.var.train.diff.diff12 = diff(diff(auto.var.train),
    lag = 12)
```

Before building the VAR model, we recall the following from our EDA:

- Our unit root tests provided strong evidence of a unit root in the unemployment series and weak evidence for that in the auto sales series, and our univariate plots exhibit noticeable seasonal patterns.
- All the Unit Root tests rejected the null hypotheses that either the first differenced, seasonally differenced, or first and seasonally differenced series for both Unemployment and Auto Sales contain a unit root, which supports stationarity.
- The auto sales appear to lead the unemployment rate by a few months. These series are negatively correlated. In addition, the univariate ACF and PACF plots for unemployment show autocorrelation with its own lags, so a VAR model for unemployment will likely include significant estimates of its own lags as well as auto sales lags.

Estimate Model Order

Now we will build a VAR model. The lag length is determined by the VARselect function. In order to test the way VAR() function works, we used four sets of data and different arguments in VAR. We will compare the model by these diagnostic tests: serial correlation, normality, and acf plots.

```
# model1,2: 1st differenced data with seasonal dummy
# variables
vars::VARselect(cbind(unem.var.train.diff, auto.var.train.diff),
    lag.max = 30, season = 12)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     12     12      3     12
##
## $criteria
##                1          2          3          4          5          6
## AIC(n)   6.077358   5.908156   5.816714   5.792824   5.779709   5.773954
## HQ(n)    6.178313   6.023533   5.946513   5.937045   5.938352   5.947019
## SC(n)    6.333475   6.200861   6.146008   6.158706   6.182179   6.213012
## FPE(n) 435.893801 368.049133 335.895625 327.976595 323.715984 321.873231
##                7          8          9         10         11         12
## AIC(n)   5.764036   5.770100   5.757378   5.752039   5.686467   5.636140
## HQ(n)    5.951523   5.972010   5.973709   5.982793   5.931643   5.895738
## SC(n)    6.239682   6.282335   6.306200   6.337450   6.308466   6.294728
## FPE(n) 318.714432 320.673837 316.643488 314.984499 295.021494 280.572002
##               13         14         15         16         17         18
## AIC(n)   5.643104   5.640704   5.652517   5.662337   5.662471   5.669571
## HQ(n)    5.917124   5.929145   5.955380   5.979623   5.994179   6.015701
## SC(n)    6.338279   6.372467   6.420868   6.467277   6.503999   6.547688
## FPE(n) 282.567032 281.927736 285.320635 288.183970 288.274650 290.386181
##               19         20         21         22         23         24
## AIC(n)   5.680029   5.669705   5.683042   5.697468   5.705056   5.706180
## HQ(n)    6.040581   6.044679   6.072438   6.101286   6.123296   6.138843
## SC(n)    6.594733   6.620997   6.670923   6.721937   6.766113   6.803826
## FPE(n) 293.501816 290.554894 294.530325 298.891002 301.255621 301.688977
##               25         26         27         28         29         30
## AIC(n)   5.710710   5.722897   5.736465   5.749737   5.763830   5.769005
## HQ(n)    6.157795   6.184403   6.212394   6.240088   6.268603   6.288200
## SC(n)    6.844944   6.893718   6.943875   6.993735   7.044416   7.086179
## FPE(n) 303.160220 306.987132 311.299587 315.586610 320.203733 322.012063
```

Using the first differenced data with seasonal dummy variables, we get different results based on the different criteria. AIC prefers a VAR(12) model, whereas BIC prefers a VAR(3).

```
# 12th differenced model3,4: deseason data with trend
# included in regression
VARselect(cbind(unem.var.train.diff12, auto.var.train.diff12),
    lag.max = 30, type = "both")
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     26     14      4     26
##
## $criteria
##                1          2          3          4          5          6
```

```
## AIC(n)   6.550910   6.520043   6.483365   6.437873   6.431812   6.432201
## HQ(n)    6.580330   6.564173   6.542205   6.511423   6.520072   6.535170
## SC(n)    6.625471   6.631884   6.632487   6.624276   6.655496   6.693165
## FPE(n) 699.881310 678.609778 654.173658 625.085691 621.315986 621.567408
##                7          8          9         10         11         12
## AIC(n)   6.447474   6.460123   6.448077   6.452240   6.462844   6.393527
## HQ(n)    6.565154   6.592512   6.595176   6.614049   6.639363   6.584756
## SC(n)    6.745719   6.795647   6.820882   6.862326   6.910210   6.878174
## FPE(n) 631.147389 639.198627 631.566728 634.228240 641.021778 598.129080
##               13         14         15         16         17         18
## AIC(n)   6.151401   6.123805   6.135997   6.147568   6.159041   6.161368
## HQ(n)    6.357341   6.344454   6.371356   6.397637   6.423820   6.440857
## SC(n)    6.673329   6.683013   6.732486   6.781337   6.830091   6.869698
## FPE(n) 469.538637 456.795045 462.440873 467.871504 473.326068 474.490998
##               19         20         21         22         23         24
## AIC(n)   6.176953   6.191358   6.200238   6.211549   6.218712   6.167520
## HQ(n)    6.471152   6.500267   6.523857   6.549878   6.571751   6.535269
## SC(n)    6.922564   6.974249   7.020410   7.069001   7.113445   7.099533
## FPE(n) 482.014173 489.086892 493.537169 499.248418 502.944629 477.956135
##               25         26         27         28         29         30
## AIC(n)   6.104903   6.072340   6.088542   6.100502   6.107836   6.116808
## HQ(n)    6.487361   6.469508   6.500421   6.527090   6.549134   6.572816
## SC(n)    7.074197   7.078914   7.132397   7.181637   7.226252   7.272504
## FPE(n) 449.058374 434.789272 442.020598 447.479433 450.925486 455.153884
```

Using the seasonally differenced data with the "trend" argument included, AIC prefers a VAR(26) model, whereas BIC prefers a VAR(4).

```
# 1st & 12th differenced model5,6: detrend & deseason
# data with no dummy variable
VARselect(cbind(unem.var.train.diff.diff12, auto.var.train.diff.diff12),
    lag.max = 30)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     25     13     12     25
##
## $criteria
##                1          2          3          4          5          6
## AIC(n)   6.792267   6.689290   6.591205   6.552788   6.540800   6.552800
## HQ(n)    6.814372   6.726132   6.642783   6.619103   6.621852   6.648589
## SC(n)    6.848285   6.782652   6.721912   6.720840   6.746197   6.795542
## FPE(n) 890.931504 803.753057 728.662009 701.204570 692.855477 701.229668
##                7          8          9         10         11         12
## AIC(n)   6.557691   6.545845   6.543894   6.548317   6.495410   6.223039
## HQ(n)    6.668217   6.671107   6.683893   6.703053   6.664883   6.407248
## SC(n)    6.837778   6.863277   6.898670   6.940439   6.924877   6.689851
## FPE(n) 704.681091 696.399541 695.063610 698.171864 662.224659 504.359380
##               13         14         15         16         17         18
## AIC(n)   6.190978   6.196872   6.208379   6.218623   6.227509   6.245167
## HQ(n)    6.389924   6.410554   6.436798   6.461779   6.485401   6.517797
## SC(n)    6.695135   6.738373   6.787225   6.834814   6.881045   6.936048
## FPE(n) 488.477430 491.401874 497.132275 502.300804 506.840434 515.934530
##               19         20         21         22         23         24
## AIC(n)   6.259514   6.266115   6.279218   6.279393   6.237198   6.178473
```

```
## HQ(n)    6.546880   6.568218   6.596058   6.610969   6.583511   6.539523
## SC(n)    6.987740   7.031686   7.082134   7.119654   7.114803   7.093424
## FPE(n) 523.462889 527.011162 534.053466 534.246843 512.278271 483.169338
##               25         26         27         28         29         30
## AIC(n)   6.141623   6.156255   6.171134   6.183556   6.192209   6.201373
## HQ(n)    6.517410   6.546778   6.576394   6.603553   6.626942   6.650843
## SC(n)    7.093919   7.145895   7.198120   7.247887   7.293884   7.340393
## FPE(n) 465.801684 472.791461 480.015237 486.163861 490.549514 495.239864
```

Using the first and seasonally differenced data without the "trend" argument included and without seasonal dummy variables, AIC prefers a VAR(25) model, whereas BIC prefers a VAR(12).

```
# raw data model7,8: raw data with trend included in
# regression and seasonal dummy variable
VARselect(cbind(unem.var.train, auto.var.train), lag.max = 30,
    type = "both", season = 12)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     13      4      4     13
##
## $criteria
##                1          2          3          4          5          6
## AIC(n)   5.955596   5.884010   5.792751   5.736120   5.731808   5.716316
## HQ(n)    6.063570   6.006381   5.929518   5.887284   5.897368   5.896272
## SC(n)    6.229546   6.194487   6.139754   6.119650   6.151865   6.172899
## FPE(n) 385.925893 359.272890 327.946908 309.901948 308.581674 303.853351
##                7          8          9         10         11         12
## AIC(n)   5.715299   5.712470   5.717751   5.717756   5.716798   5.645467
## HQ(n)    5.909652   5.921220   5.940897   5.955298   5.968737   5.911802
## SC(n)    6.208409   6.242107   6.283914   6.320445   6.356014   6.321210
## FPE(n) 303.562558 302.725999 304.352924 304.381877 304.121591 283.216273
##               13         14         15         16         17         18
## AIC(n)   5.613129   5.620887   5.621708   5.633582   5.640832   5.638495
## HQ(n)    5.893861   5.916015   5.931233   5.957503   5.979150   5.991209
## SC(n)    6.325398   6.369683   6.407031   6.455431   6.499208   6.533397
## FPE(n) 274.239200 276.414214 276.684368 280.037424 282.128007 281.527002
##               19         20         21         22         23         24
## AIC(n)   5.644440   5.655061   5.649709   5.663132   5.675042   5.683417
## HQ(n)    6.011551   6.036569   6.045613   6.073433   6.099739   6.122510
## SC(n)    6.575870   6.623017   6.654192   6.704142   6.752578   6.797479
## FPE(n) 283.268788 286.362313 284.907811 288.838660 292.387163 294.940892
##               25         26         27         28         29         30
## AIC(n)   5.681346   5.690357   5.702597   5.716520   5.729388   5.743720
## HQ(n)    6.134836   6.158244   6.184881   6.213199   6.240464   6.269193
## SC(n)    6.831936   6.877473   6.926240   6.976689   7.026084   7.076943
## FPE(n) 294.432166 297.206300 300.984183 305.330690 309.421621 314.034847
```

Using the raw time series data without any differencing, but including the "trend" argument and seasonal dummy variables, AIC prefers a VAR(13) model, whereas BIC prefers a VAR(4).

Notice that the above VARselect results suggest very different numbers of lags to include by AICc vs BIC. This is because the two information criteria penalize by different terms. The R documentation shows the following formulation for AICc and BIC.

$$AIC(n) = \ln \hat{\sigma}_k^2 + [2k + 2k(k+1)/(n-k-1)]$$

$$SC(n) = \ln \hat{\sigma}_k^2 + kIn(n)$$

While the first component in each formula are the same and refer to the log of sum square error, the second terms–the penalty terms–are different. In general, as sample size n grows, BIC's penalty term grows faster than AICc's penalty term with number of estimated lags k. Therefore, BIC is better at penalizing large samples more consistenly and heavily. Considering our sample size of 479, we should choose the orders estimated by BIC over AICc, if both model residuals exhibit white noise behavior. The estimated orders are:

- Both series first differenced and seasonal variable(model 1, 2) : (Order 3 by BIC) (Order 12 by AIC)
- Both series seasonal differenced with trend included in regression(model 3, 4): (Order 4 by BIC) (Order 26 by AIC)
  - Recall from EDA that peak of cross-correlation took place between lag 3-7
- Both series first and seasonal differenced with no dummy variable (model 5, 6) : (Order 12 by BIC) (Order 25 by AIC)
- Both series raw with trend as regressor and seasonal dummy variable (model 7, 8) : (Order 4 by BIC) (Order 13 by AIC)

Next we will fit the eight models discussed above and examine the residuals.

```
# model1,2: 1st differenced data with season dummy
# variable
var.diff.mod.lag3 = VAR(cbind(unem.var.train.diff, auto.var.train.diff),
    p = 3, season = 12)

var.diff.mod.lag12 = VAR(cbind(unem.var.train.diff, auto.var.train.diff),
    p = 12, season = 12)

# 12th differenced model3,4: deseason data with trend
# included in regression
var.diff12.mod.lag4 = vars::VAR(cbind(unem.var.train.diff12,
    auto.var.train.diff12), p = 4, type = "both")

var.diff12.mod.lag26 = vars::VAR(cbind(unem.var.train.diff12,
    auto.var.train.diff12), p = 26, type = "both")

# 1st & 12th differenced model5,6: detrend & deseason
# data with no dummy variable
var.diff.diff12.mod.lag12 = VAR(cbind(unem.var.train.diff.diff12,
    auto.var.train.diff.diff12), p = 12)

var.diff.diff12.mod.lag25 = VAR(cbind(unem.var.train.diff.diff12,
    auto.var.train.diff.diff12), p = 25)
# raw data model7,8: raw data with trend included in
# regression and seasonal dummy variable
var.mod.lag4 = VAR(cbind(unem.var.train, auto.var.train),
    p = 4, type = "both", season = 12)
var.mod.lag13 = VAR(cbind(unem.var.train, auto.var.train),
    p = 13, type = "both", season = 12)
```

Diagnostic Testing – Serial Correlation

We perform the Portmanteau Test on the model residuals to detect autocorrelation. The test hypothesis is as follows:

- Ho : The residuals are not serially correlated.
- Ha : The residuals are serially correlated.

```
# model1,2: 1st differenced data with season dummy
# variable
serial.test(var.diff.mod.lag3, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff.mod.lag3
## Chi-squared = 260.03, df = 108, p-value = 1.477e-14
```

```
serial.test(var.diff.mod.lag12, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff.mod.lag12
## Chi-squared = 77.635, df = 72, p-value = 0.3039
```

For the first differenced data with seasonal dummy variables, the VAR(3) model rejects the null hypothesis that the residuals are not serially correlated, while the VAR(12) model fails to reject.

```
# 12th differenced model3,4: deseason data with trend
# included in regression
serial.test(var.diff12.mod.lag4, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff12.mod.lag4
## Chi-squared = 269.53, df = 104, p-value < 2.2e-16
```

```
serial.test(var.diff12.mod.lag26, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff12.mod.lag26
## Chi-squared = 47.093, df = 16, p-value = 6.586e-05
```

For the seasonally differenced data with trend included, the VAR(4) and VAR(26) models both reject the null hypothesis that the residuals are not serially correlated.

```
# 1st & 12th differenced model5,6: detrend & deseason
# data with no dummy variable
serial.test(var.diff.diff12.mod.lag12, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag12
## Chi-squared = 163.79, df = 72, p-value = 4.162e-09
```

```
serial.test(var.diff.diff12.mod.lag25, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
```

```
## data:  Residuals of VAR object var.diff.diff12.mod.lag25
## Chi-squared = 53.557, df = 20, p-value = 6.729e-05
```

For the first and seasonally differenced data without trend or seasonal dummy variables included, the VAR(12) and VAR(25) models both reject the null hypothesis that the residuals are not serially correlated.

```
# raw data model7,8: raw data with trend included in
# regression and seasonal dummy variable
serial.test(var.mod.lag4, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.mod.lag4
## Chi-squared = 236.87, df = 104, p-value = 2.297e-12
```

```
serial.test(var.mod.lag13, lags.pt = 30)
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.mod.lag13
## Chi-squared = 71.77, df = 68, p-value = 0.3541
```

For the raw data with trend and seasonal dummy variables included, the VAR(4) model rejects the null hypothesis that the residuals are not serieally correlated, but the VAR(13) model fails to reject this null hypothesis.

According to the p-values, Model 2 (a VAR(12) model on first differenced data with seasonal dummy variables) and Model 7 (a VAR(13) model on the raw data with trend and seasonal dummy variables included) failed to reject the null hypothesis that there's no serial correlation in the residuals, so these may be good candidates.

Diagnostic Testing – Normality

Next we perform the normality test on the model residuals to detect autocorrelation. Test hypothesis is as follows:

- Ho : The residuals are normally distributed
- Ha : The residuals are NOT normally distributed

```
normality.test(var.diff.mod.lag3, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag3
## Chi-squared = 127, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag3
## Chi-squared = 17.83, df = 2, p-value = 0.0001343
##
```

```
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag3
## Chi-squared = 109.17, df = 2, p-value < 2.2e-16
```

```
normality.test(var.diff.mod.lag12, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag12
## Chi-squared = 150, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag12
## Chi-squared = 16.266, df = 2, p-value = 0.0002937
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff.mod.lag12
## Chi-squared = 133.74, df = 2, p-value < 2.2e-16
```

For the first differenced data with seasonal dummy variables, both the VAR(3) and the VAR(12) models reject the null hypothesis that the residuals are normally distributed.

```
normality.test(var.diff12.mod.lag4, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag4
## Chi-squared = 155.97, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag4
## Chi-squared = 3.2758, df = 2, p-value = 0.1944
##
##
## $Kurtosis
##
```

```
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag4
## Chi-squared = 152.7, df = 2, p-value < 2.2e-16
```

```
normality.test(var.diff12.mod.lag26, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag26
## Chi-squared = 58.888, df = 4, p-value = 4.968e-12
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag26
## Chi-squared = 3.3295, df = 2, p-value = 0.1892
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff12.mod.lag26
## Chi-squared = 55.558, df = 2, p-value = 8.624e-13
```

For the seasonally differenced data with trend included, both the VAR(4) and the VAR(26) models fail to reject the null hypothesis that the residuals are normally distributed.

```
normality.test(var.diff.diff12.mod.lag12, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag12
## Chi-squared = 130.19, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag12
## Chi-squared = 2.8654, df = 2, p-value = 0.2387
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag12
```

```
## Chi-squared = 127.32, df = 2, p-value < 2.2e-16
```

```
normality.test(var.diff.diff12.mod.lag25, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag25
## Chi-squared = 59.711, df = 4, p-value = 3.336e-12
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag25
## Chi-squared = 2.7104, df = 2, p-value = 0.2579
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag25
## Chi-squared = 57.001, df = 2, p-value = 4.192e-13
```

For the first and seasonally differenced data, the VAR(12) and VAR(25) models both fail to reject the null hypothesis of normality.

```
normality.test(var.mod.lag4, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.mod.lag4
## Chi-squared = 159.56, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.mod.lag4
## Chi-squared = 29.162, df = 2, p-value = 4.65e-07
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.mod.lag4
## Chi-squared = 130.4, df = 2, p-value < 2.2e-16
```

```r
normality.test(var.mod.lag13, multivariate.only = TRUE)
```

```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object var.mod.lag13
## Chi-squared = 194.4, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object var.mod.lag13
## Chi-squared = 23.144, df = 2, p-value = 9.425e-06
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object var.mod.lag13
## Chi-squared = 171.26, df = 2, p-value < 2.2e-16
```

Finally, for the raw data with trend and seasonal dummy variables included, the VAR(4) and VAR(13) models both reject the null hypothesis of normality.

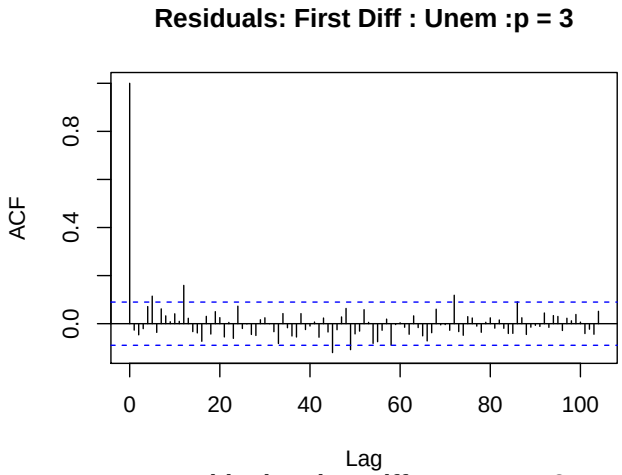
From these tests, the following models did not reject the null hypothesis of normally distributed residuals and may be good candidates:

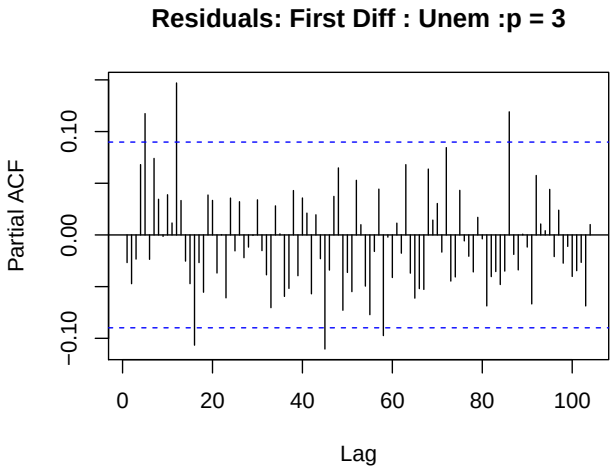
- Model 3: VAR(4) on seasonally differenced data with trend included
- Model 4: VAR(26) on seasonally differenced data with trend included
- Model 5: VAR(12) on first and seasonally differenced data
- Model 6: VAR(25) on first and seasonally differenced data

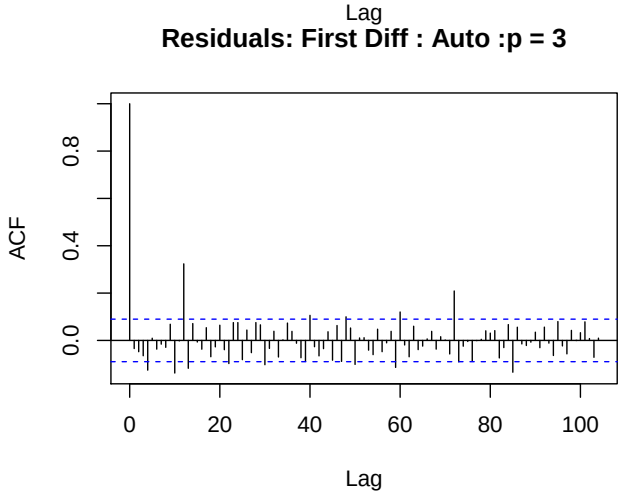
Diagnostic Testing – Residual ACF Plots

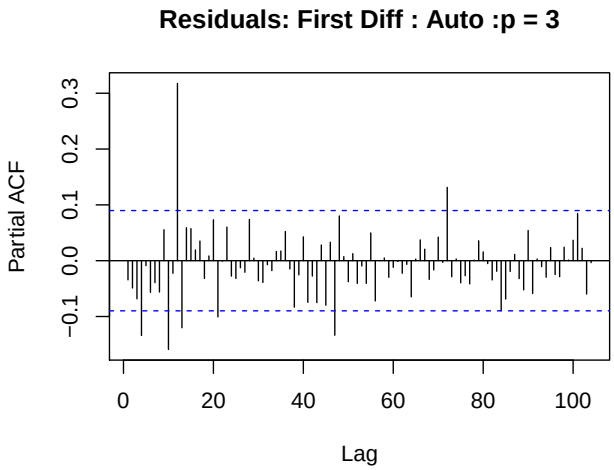
Next we will examine the ACF and PACF plots of the residuals for each model.

```r
# 1st differenced & seasonal dummies p = 3, model1
acf(residuals(var.diff.mod.lag3)[, 1], 104, main = "")
title("Residuals: First Diff : Unem :p = 3")
pacf(residuals(var.diff.mod.lag3)[, 1], 104, main = "")
title("Residuals: First Diff : Unem :p = 3")

acf(residuals(var.diff.mod.lag3)[, 2], 104, main = "")
title("Residuals: First Diff : Auto :p = 3")
pacf(residuals(var.diff.mod.lag3)[, 2], 104, main = "")
title("Residuals: First Diff : Auto :p = 3")
```



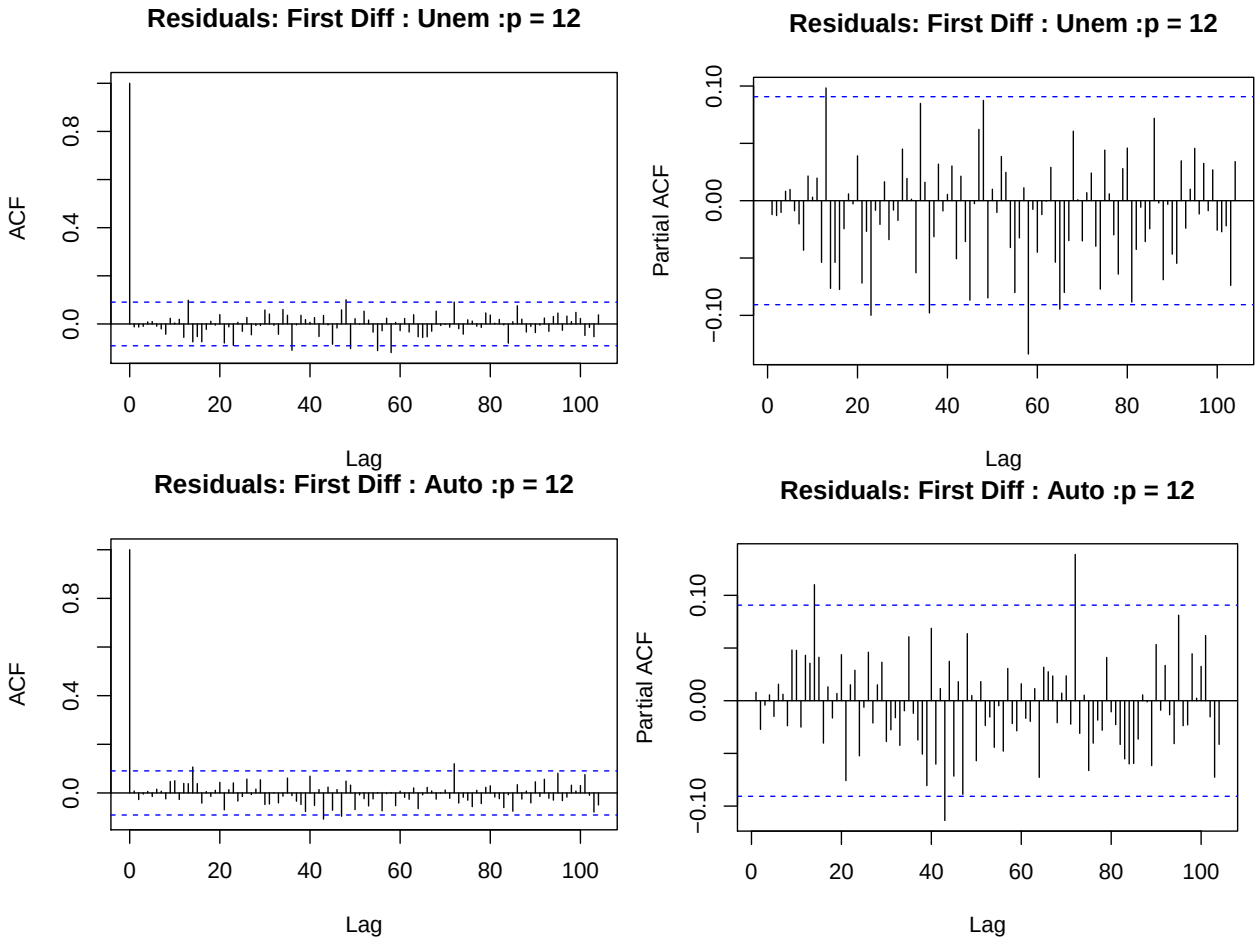






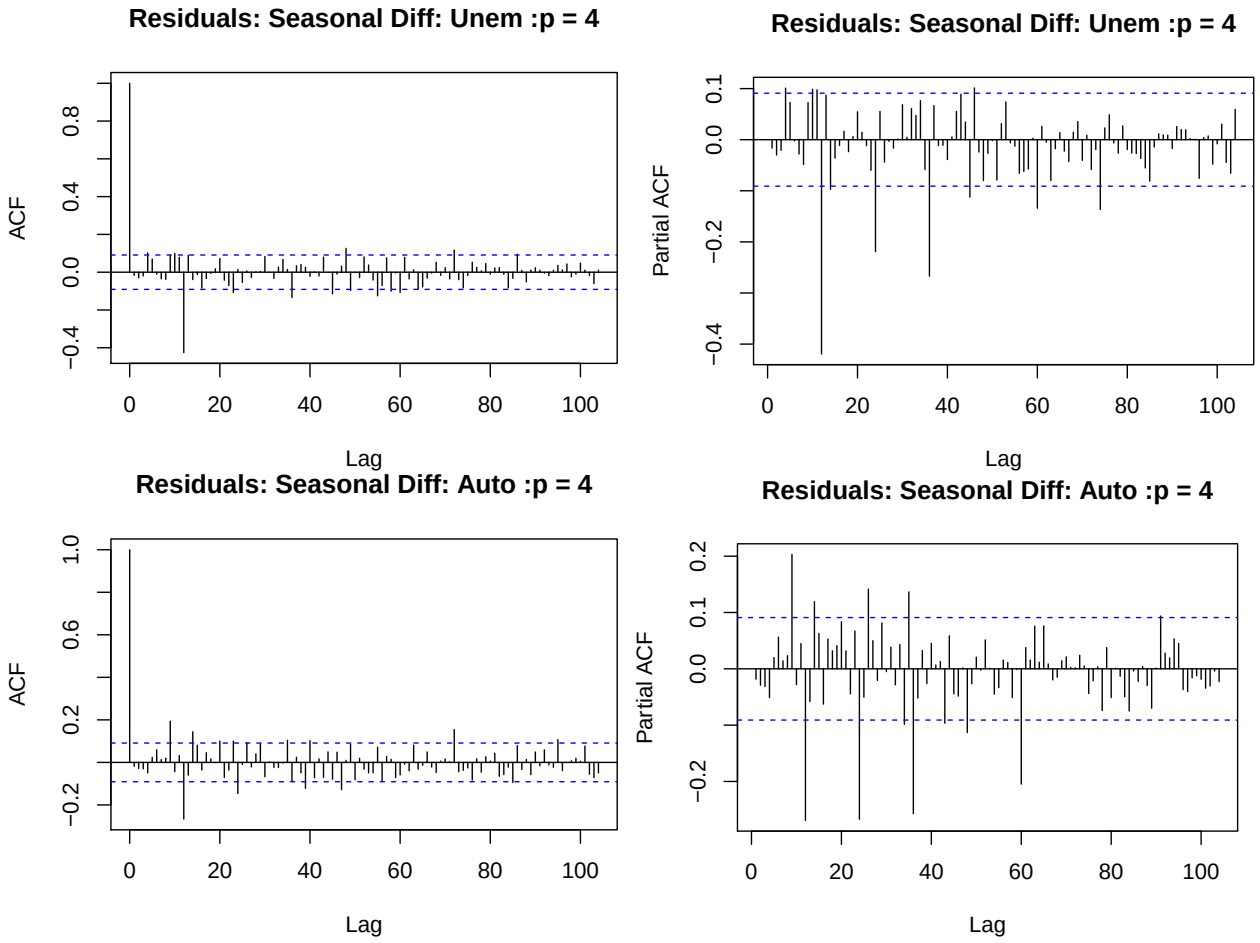
For the VAR(3) model on first differenced data with seasonal dummy variables, the ACF and PACF plots for both unemployment and auto sales show a few significant spikes.

```
# 1st differenced & seasonal dummies p = 12, model2
acf(residuals(var.diff.mod.lag12)[, 1], 104, main = "")
title("Residuals: First Diff : Unem :p = 12")
pacf(residuals(var.diff.mod.lag12)[, 1], 104, main = "")
title("Residuals: First Diff : Unem :p = 12")

acf(residuals(var.diff.mod.lag12)[, 2], 104, main = "")
title("Residuals: First Diff : Auto :p = 12")
pacf(residuals(var.diff.mod.lag12)[, 2], 104, main = "")
title("Residuals: First Diff : Auto :p = 12")
```



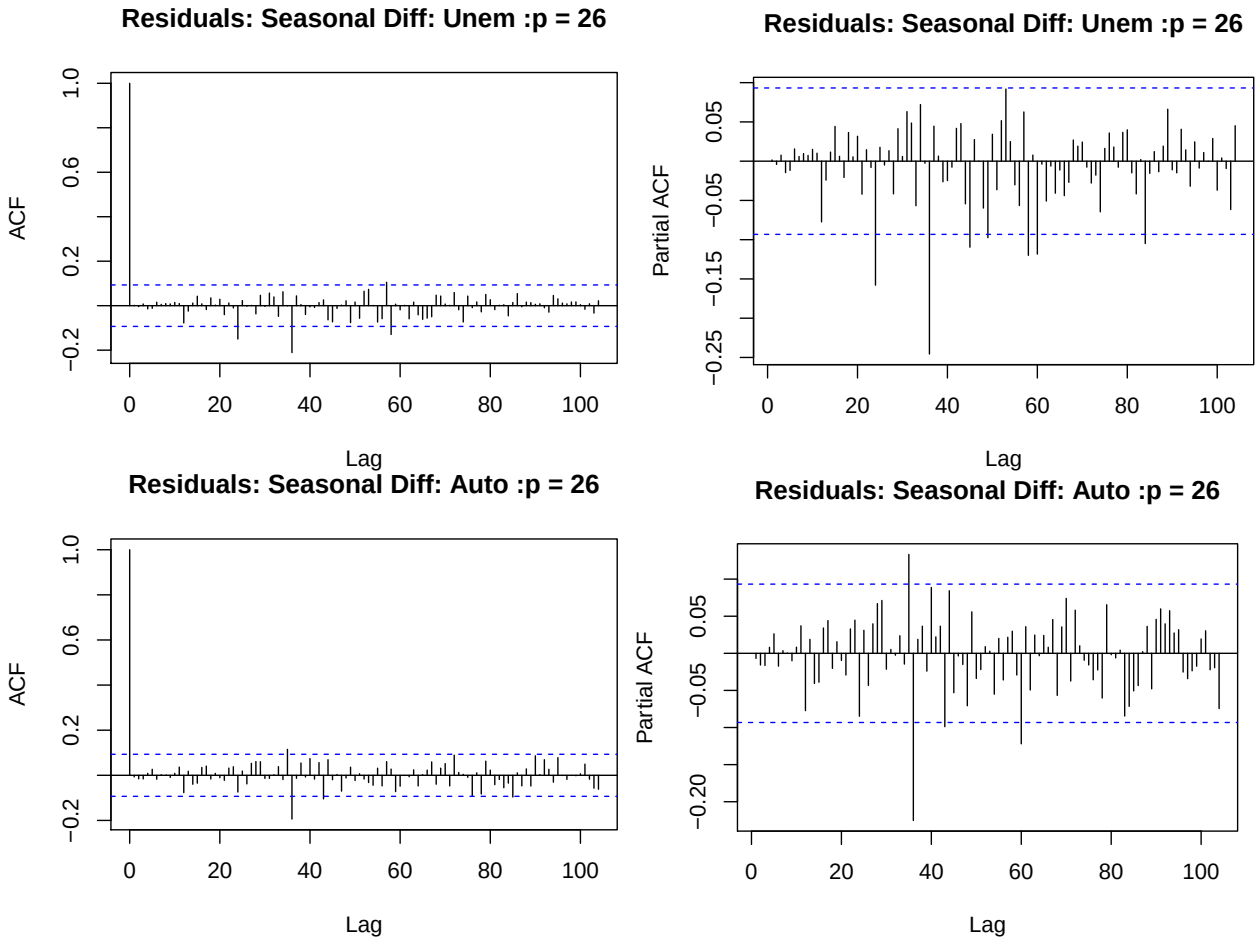
For the VAR(12) model on first differenced data with seasonal dummy variables, the ACF and PACF plots for Unemployment and Auto Sales fairly closely resemble white noise, although there are still a few significant spikes.

```
# seasonal differenced p = 4, model3
acf(residuals(var.diff12.mod.lag4)[, 1], 104, main = "")
title("Residuals: Seasonal Diff: Unem :p = 4")
pacf(residuals(var.diff12.mod.lag4)[, 1], 104, main = "")
title("Residuals: Seasonal Diff: Unem :p = 4")

acf(residuals(var.diff12.mod.lag4)[, 2], 104, main = "")
title("Residuals: Seasonal Diff: Auto :p = 4")
pacf(residuals(var.diff12.mod.lag4)[, 2], 104, main = "")
title("Residuals: Seasonal Diff: Auto :p = 4")
```



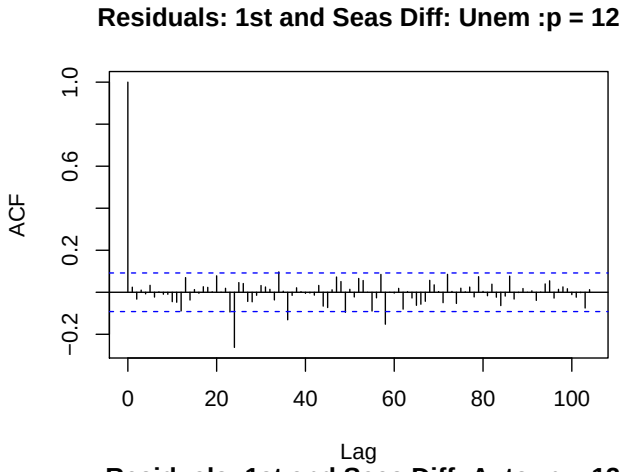
For the VAR(4) model on seasonally differenced data with trend, the ACF and PACF plots for both series have several significant spikes and do not resemble white noise.

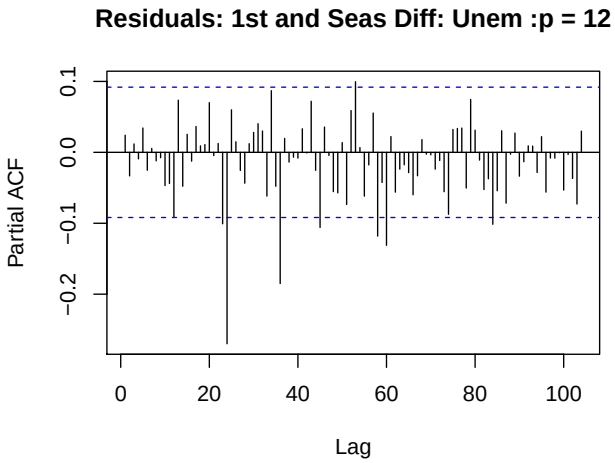
```
# seasonal differenced p = 26, model4
acf(residuals(var.diff12.mod.lag26)[, 1], 104, main = "")
title("Residuals: Seasonal Diff: Unem :p = 26")
pacf(residuals(var.diff12.mod.lag26)[, 1], 104, main = "")
title("Residuals: Seasonal Diff: Unem :p = 26")

acf(residuals(var.diff12.mod.lag26)[, 2], 104, main = "")
title("Residuals: Seasonal Diff: Auto :p = 26")
pacf(residuals(var.diff12.mod.lag26)[, 2], 104, main = "")
title("Residuals: Seasonal Diff: Auto :p = 26")
```

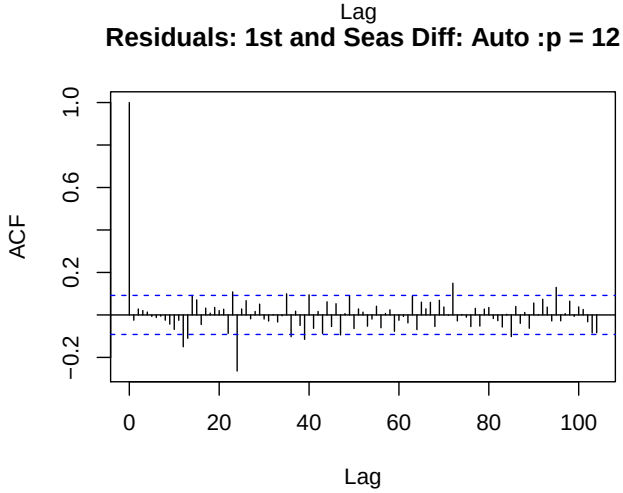


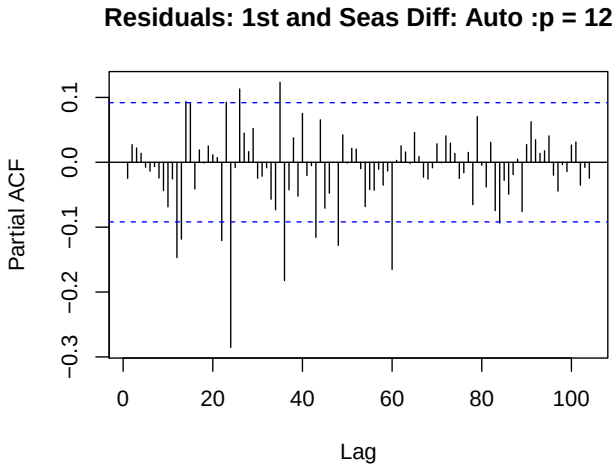
For the VAR(26) model on seasonally differenced data with trend, the ACF and PACF plots have some very significant spikes and do not resemble white noise.

```
# 1st differenced & seasonal differenced p = 12, model5
acf(residuals(var.diff.diff12.mod.lag12)[, 1], 104, main = "")
title("Residuals: 1st and Seas Diff: Unem :p = 12")
pacf(residuals(var.diff.diff12.mod.lag12)[, 1], 104, main = "")
title("Residuals: 1st and Seas Diff: Unem :p = 12")

acf(residuals(var.diff.diff12.mod.lag12)[, 2], 104, main = "")
title("Residuals: 1st and Seas Diff: Auto :p = 12")
pacf(residuals(var.diff.diff12.mod.lag12)[, 2], 104, main = "")
title("Residuals: 1st and Seas Diff: Auto :p = 12")
```



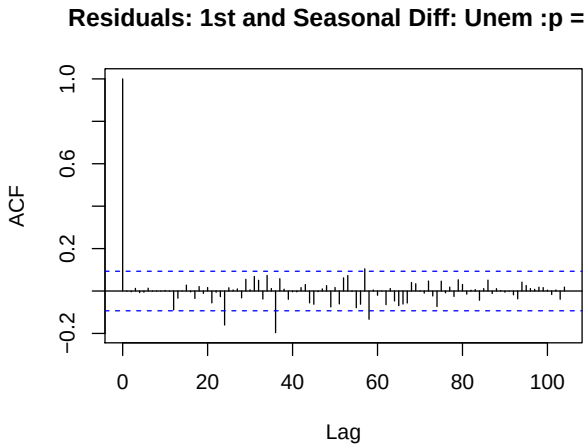


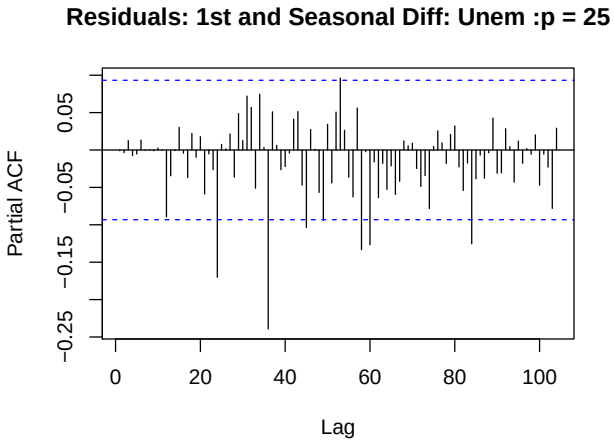


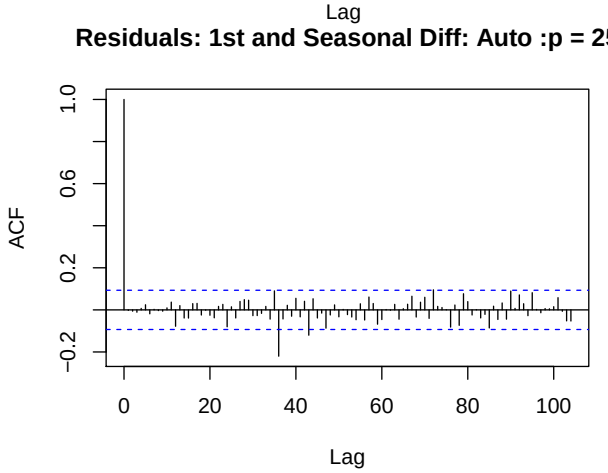


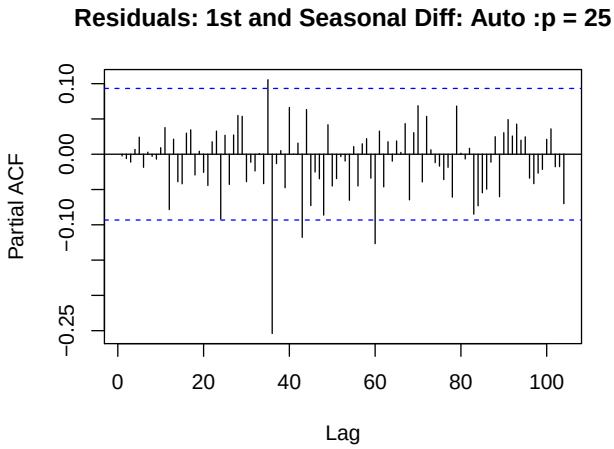
For the VAR(12) model on first and seasonally differenced data, the ACF and PACF plots for both series have significant spikes and do not resemble white noise.

```
# 1st differenced & seasonal differenced p = 25, model6
acf(residuals(var.diff.diff12.mod.lag25)[, 1], 104, main = "")
title("Residuals: 1st and Seasonal Diff: Unem :p = 25")
pacf(residuals(var.diff.diff12.mod.lag25)[, 1], 104, main = "")
title("Residuals: 1st and Seasonal Diff: Unem :p = 25")

acf(residuals(var.diff.diff12.mod.lag25)[, 2], 104, main = "")
title("Residuals: 1st and Seasonal Diff: Auto :p = 25")
pacf(residuals(var.diff.diff12.mod.lag25)[, 2], 104, main = "")
title("Residuals: 1st and Seasonal Diff: Auto :p = 25")
```



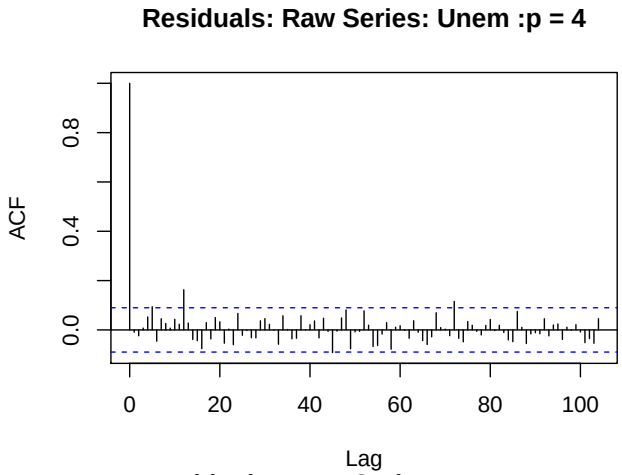


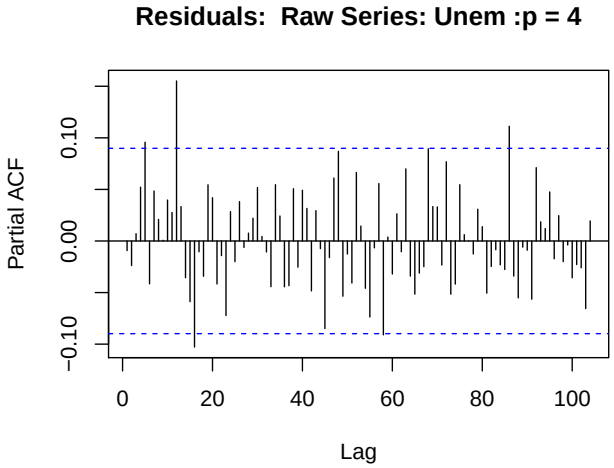


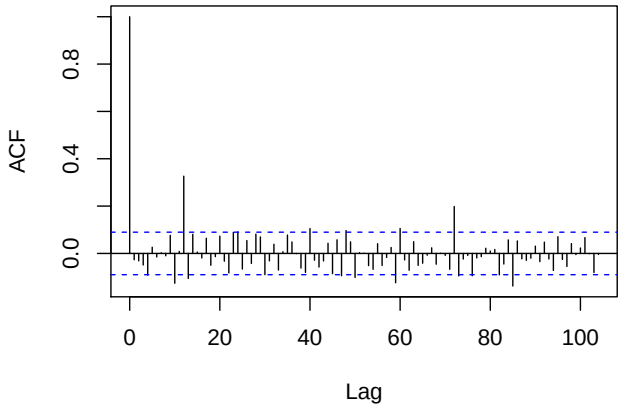


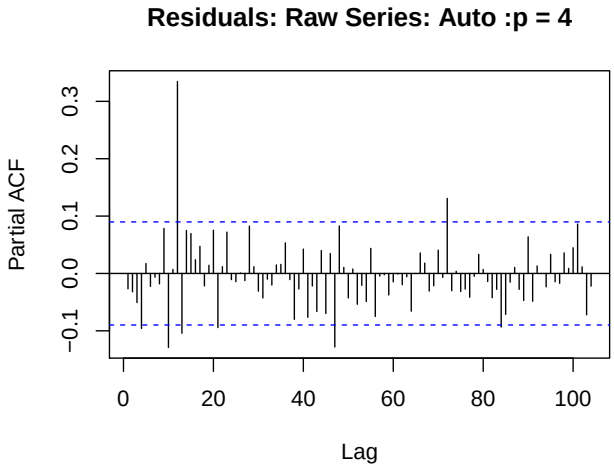
For the VAR(25) model on first and seasonally differenced data, the ACF and PACF plots for both series have significant spikes and do not resemble white noise.

```
# raw data with trend included in regression and
# seasonal dummy variable p = 4, model7
acf(residuals(var.mod.lag4)[, 1], 104, main = "")
title("Residuals: Raw Series: Unem :p = 4")
pacf(residuals(var.mod.lag4)[, 1], 104, main = "")
title("Residuals:  Raw Series: Unem :p = 4")

acf(residuals(var.mod.lag4)[, 2], 104, main = "")
title("Residuals:  Raw Series: Auto :p = 4")
pacf(residuals(var.mod.lag4)[, 2], 104, main = "")
title("Residuals: Raw Series: Auto :p = 4")
```



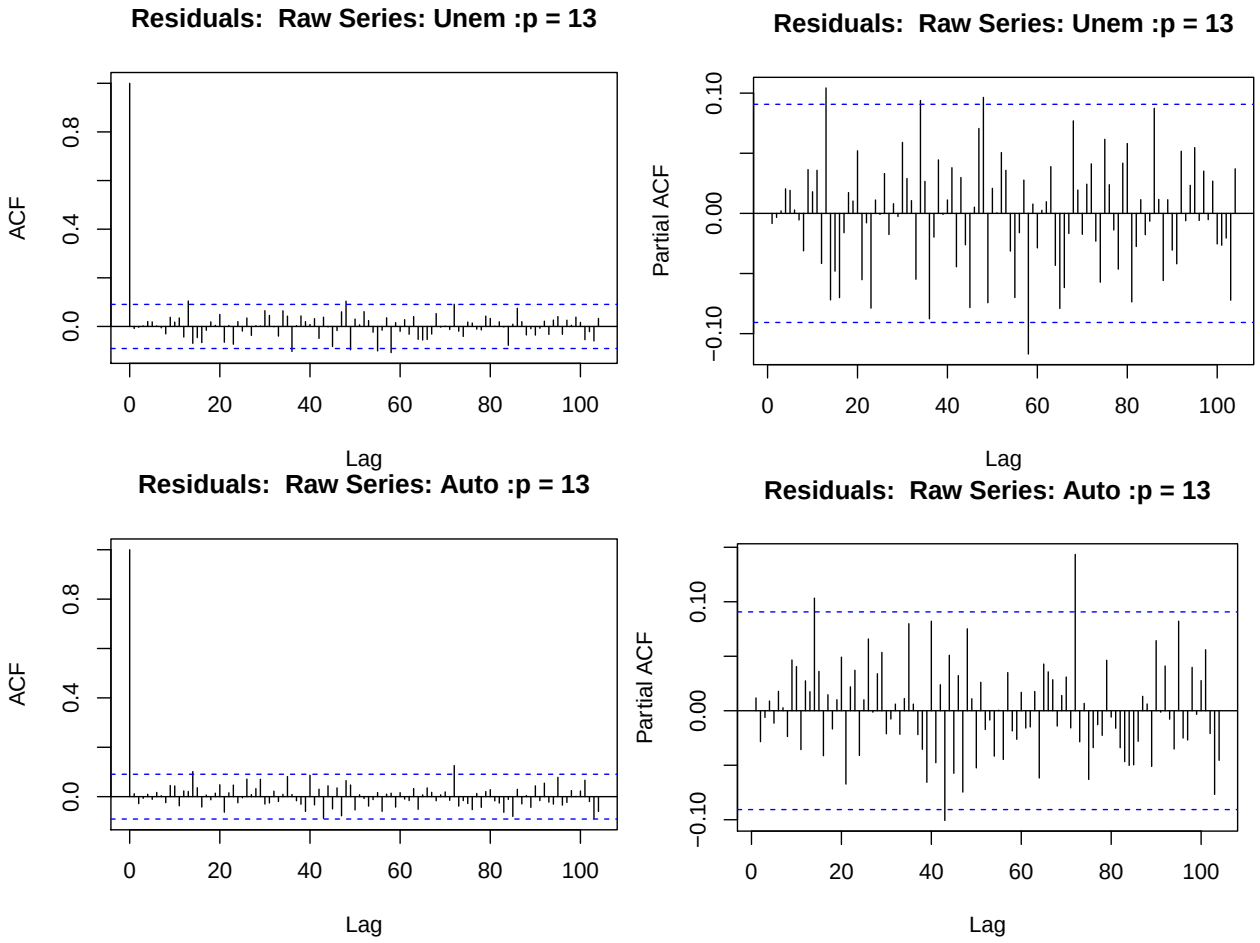






For the VAR(4) model on the raw data with trend and seasonal dummy variables, the ACF and PACF plots both have some very significant spikes and do not resemble white noise.

```
# raw data with trend included in regression and
# seasonal dummy variable p = 13, model8
acf(residuals(var.mod.lag13)[, 1], 104, main = "")
title("Residuals:  Raw Series: Unem :p = 13")
pacf(residuals(var.mod.lag13)[, 1], 104, main = "")
title("Residuals:  Raw Series: Unem :p = 13")

acf(residuals(var.mod.lag13)[, 2], 104, main = "")
title("Residuals:  Raw Series: Auto :p = 13")
pacf(residuals(var.mod.lag13)[, 2], 104, main = "")
title("Residuals:  Raw Series: Auto :p = 13")
```



For the VAR(13) model on the raw data with trend and seasonal dummy variables included, the ACF and PACF plots have a few significant and near significant spikes and do not closely resemble white noise.

From the residual plots above, the model with first differenced series and seasonal variables performed better in general. Lag order 12 gives residuals that are closest to white noise. We reduced it down to lag order 11 with similar white noise patterns, but the Portmanteau Test (below) rejected the null hypothesis that the residuals are not serially correlated. A model of lag order 10 and 9 starts to show significant pacfs at lag 12 therefore we stop searching down from there. Time plots and residual plots for lag order 11 and 10 are given below as well.

```
var.diff.mod.lag11 = vars::VAR(cbind(unem.var.train.diff,
    auto.var.train.diff), p = 11, season = 12)
vars::serial.test(var.diff.mod.lag11, lags.pt = 30, type = "PT.adjusted")
```

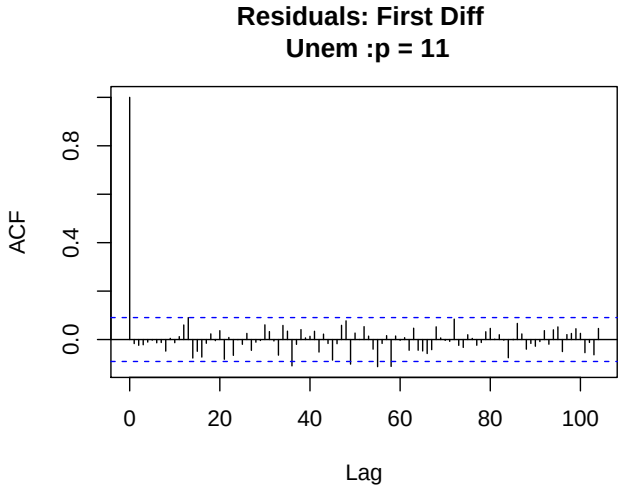
```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.mod.lag11
## Chi-squared = 101.01, df = 76, p-value = 0.02914
```

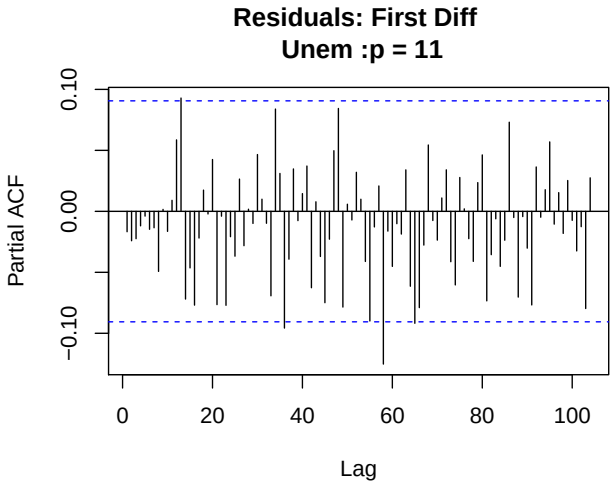
```
var.diff.mod.lag10 = vars::VAR(cbind(unem.var.train.diff,
    auto.var.train.diff), p = 10, season = 12)
vars::serial.test(var.diff.mod.lag10, lags.pt = 30, type = "PT.adjusted")
```

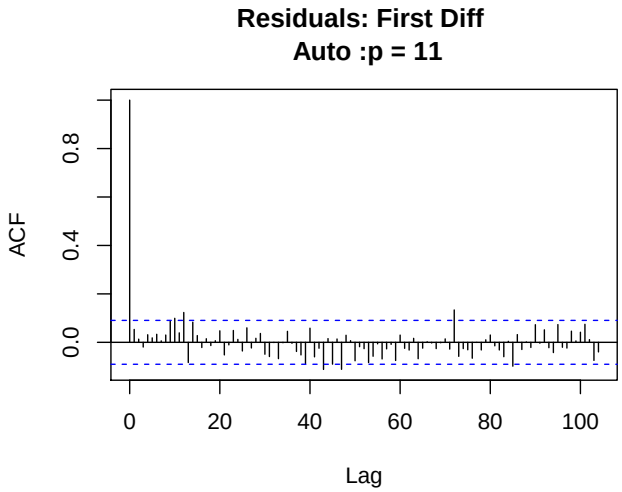
```
##
##  Portmanteau Test (adjusted)
```

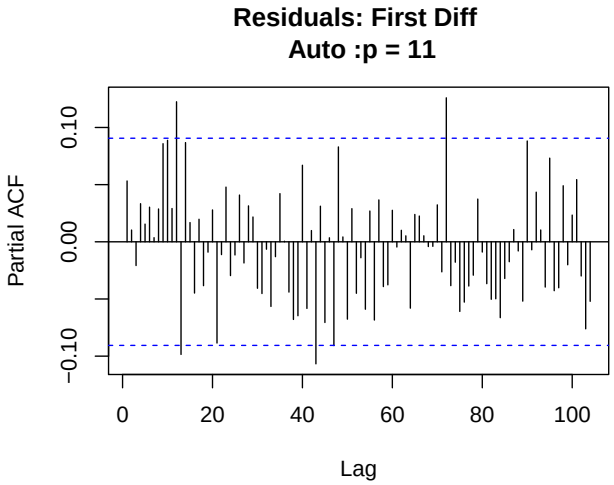
```
##
## data:  Residuals of VAR object var.diff.mod.lag10
## Chi-squared = 135.61, df = 80, p-value = 0.000104
```

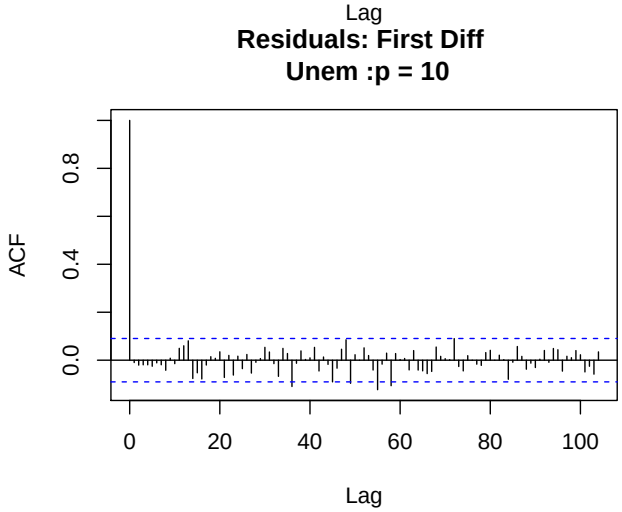
```r
# 1st differenced & seasonal dummies p = 11
acf(residuals(var.diff.mod.lag11)[, 1], 104, main = "")
title("Residuals: First Diff \n Unem :p = 11")
pacf(residuals(var.diff.mod.lag11)[, 1], 104, main = "")
title("Residuals: First Diff \n Unem :p = 11")

acf(residuals(var.diff.mod.lag11)[, 2], 104, main = "")
title("Residuals: First Diff \n Auto :p = 11")
pacf(residuals(var.diff.mod.lag11)[, 2], 104, main = "")
title("Residuals: First Diff \n Auto :p = 11")

# p = 10
acf(residuals(var.diff.mod.lag10)[, 1], 104, main = "")
title("Residuals: First Diff \n Unem :p = 10")
pacf(residuals(var.diff.mod.lag10)[, 1], 104, main = "")
title("Residuals: First Diff \n Unem :p = 10")

acf(residuals(var.diff.mod.lag10)[, 2], 104, main = "")
title("Residuals: First Diff \n Auto :p = 10")
pacf(residuals(var.diff.mod.lag10)[, 2], 104, main = "")
title("Residuals: First Diff \n Auto :p = 10")
```

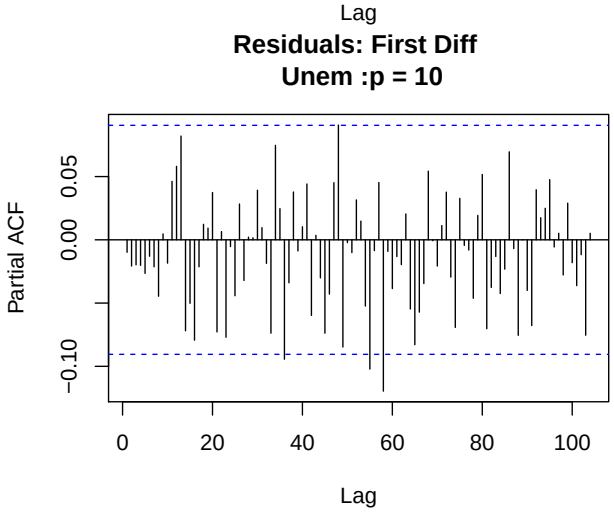


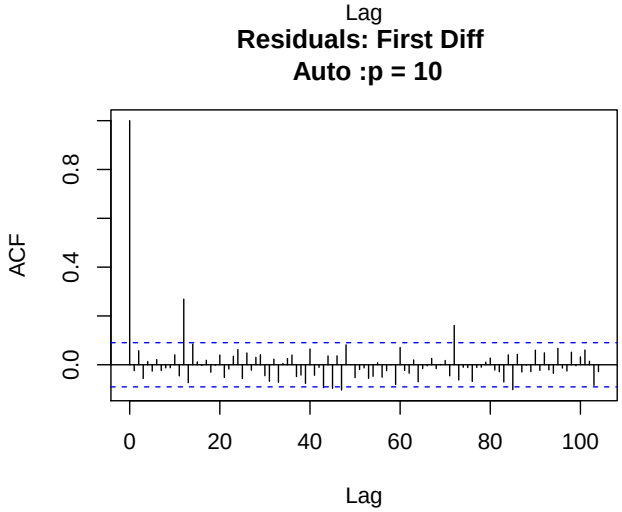


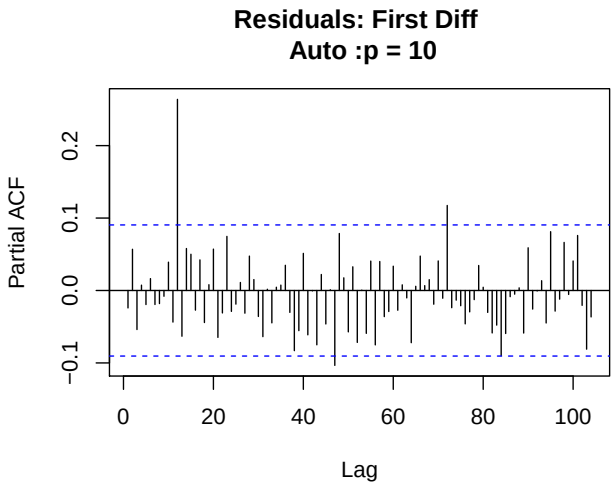












Based on the Portmanteau Tests and residual results, we proceed with the following three models. In all three, both raw series are first differenced and seasonal variables (as indicators) are included in the VAR model.

- VAR(3), VAR(11) and VAR(12)

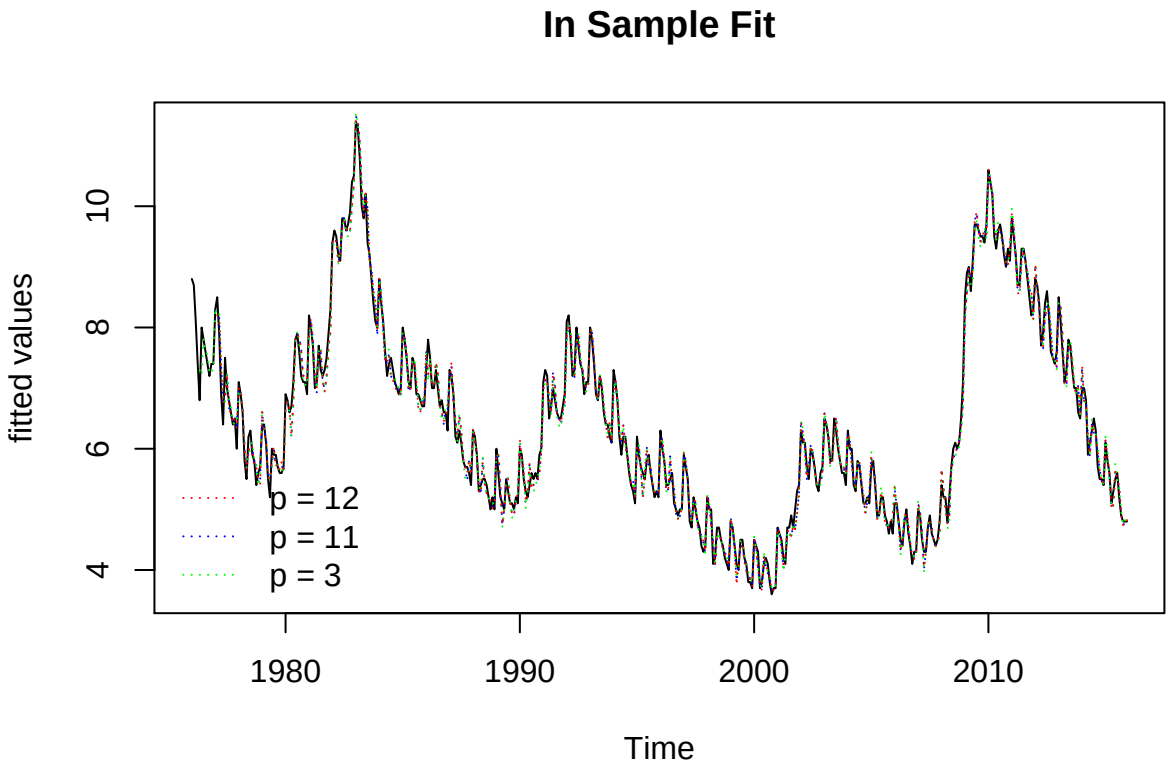
Examine In-Sample Fit

First, we define a function to compute the appropriate predictions from the VAR model, since we used first differencing.

```
# Get in-sample-fit by manually adding differenced
# estimates
get_insampfit_delta = function(mod, train) {
    na.pad = length(train) - length(fitted(mod)[, 1])
    delta <- c(rep(NA, na.pad), fitted(mod)[, 1])
    insampfit = rep(NA, length(train))
    for (i in (na.pad + 1):480) insampfit[i] <- train[i -
        1] + delta[i]
    insampfit
}
```

```
insampfit.lag12 = get_insampfit_delta(var.diff.mod.lag12,
    unem.var.train)
insampfit.lag11 = get_insampfit_delta(var.diff.mod.lag11,
    unem.var.train)
insampfit.lag3 = get_insampfit_delta(var.diff.mod.lag3,
    unem.var.train)

insampfit.lag12 = ts(insampfit.lag12, start = c(1976, 1),
    frequency = 12)
insampfit.lag11 = ts(insampfit.lag11, start = c(1976, 1),
    frequency = 12)
insampfit.lag3 = ts(insampfit.lag3, start = c(1976, 1),
    frequency = 12)
```

```
plot(unem.var.train, main = "", ylab = "fitted values")
lines(insampfit.lag11, col = "blue", lty = "dotted")
lines(insampfit.lag3, col = "green", lty = "dotted")
lines(insampfit.lag12, col = "red", lty = "dotted")
title("In Sample Fit")

legend("bottomleft", legend = c("p = 12", "p = 11", "p = 3"),
    lty = c("dotted", "dotted", "dotted"), bty = "n", col = c("red",
        "blue", "green"))
```



The in-sample fit for all three models are very close to the raw series. To further distinguish between them, we examine their out-of-sample performance next.

Compare Models Based on Out of Sample Errors Through Jun 2017

Since the VAR() function does not integrate the model for forecasting like Arima() does, and since we used first differencing, we define a function to compute the appropriate forecasts from the VAR model.

```
# Get out-of-sample-performance by manually adding
# differenced estimates set X_t-1 obs as base
base = window(combined.raw, start = c(2015, 12))[, 1]
# Function
get_outsamp = function(mod, base, ahead) {
    pred = predict(mod, n.ahead = ahead)$fcst$unem.var.train.diff
    delta.estimate = pred[, 1]
    lower.ci = pred[, 2]
    upper.ci = pred[, 3]
    var.fore = rep(0, 18)
    var.low.ci = rep(0, 18)
    var.upp.ci = rep(0, 18)
    for (i in 1:18) var.fore[i] <- base[i] + delta.estimate[i]
    for (i in 1:18) var.low.ci[i] <- base[i] + lower.ci[i]
    for (i in 1:18) var.upp.ci[i] <- base[i] + upper.ci[i]
    data.frame(var.fore, var.low.ci, var.upp.ci)
}
```

Next we compute forecasts for the next 18 months from the VAR(3), VAR(11), and VAR(12) models.

```
f.var.3 = get_outsamp(var.diff.mod.lag3, base, 18)
f.var.11 = get_outsamp(var.diff.mod.lag11, base, 18)
```

```
f.var.12 = get_outsamp(var.diff.mod.lag12, base, 18)
```

```
var.fore.df = data.frame(year = as.numeric(floor(time(unem.var.test))),
    month = c(seq(1, 12), seq(1, 6)), VAR3.est = f.var.3$var.fore,
    VAR3.err = f.var.3$var.fore - as.numeric(unem.var.test),
    VAR11.est = f.var.11$var.fore, VAR11.err = f.var.11$var.fore -
        as.numeric(unem.var.test), VAR12.est = f.var.12$var.fore,
    VAR12.err = f.var.12$var.fore - as.numeric(unem.var.test))

var.fore.df
```

```
##    year month VAR3.est     VAR3.err VAR11.est    VAR11.err VAR12.est
## 1  2016     1 5.621538  0.321538258  5.535232  0.235232221  5.539000
## 2  2016     2 5.170461 -0.029539481  5.129731 -0.070268514  5.103836
## 3  2016     3 4.945519 -0.154481485  4.922263 -0.177736786  4.912951
## 4  2016     4 4.591826 -0.108174249  4.512829 -0.187171475  4.507045
## 5  2016     5 4.650086  0.150086482  4.694008  0.194007781  4.696482
## 6  2016     6 4.971358 -0.128642488  4.818766 -0.281233625  4.792894
## 7  2016     7 5.049099 -0.050901065  5.062902 -0.037097732  5.062423
## 8  2016     8 4.861781 -0.138219150  4.829834 -0.170166461  4.798711
## 9  2016     9 4.817715  0.017715031  4.784713 -0.015286714  4.764793
## 10 2016    10 4.672266 -0.027733920  4.662989 -0.037010867  4.658773
## 11 2016    11 4.770360  0.370359571  4.725756  0.325755722  4.705915
## 12 2016    12 4.394766 -0.105234304  4.451277 -0.048723444  4.427971
## 13 2017     1 5.392585  0.292584804  5.347755  0.247755277  5.341262
## 14 2017     2 4.994263  0.094262624  4.981403  0.081403130  4.972461
## 15 2017     3 4.648712  0.048711942  4.633748  0.033747730  4.634662
## 16 2017     4 4.092220 -0.007780154  4.082557 -0.017442921  4.069027
## 17 2017     5 4.066310 -0.033689518  4.089880 -0.010120199  4.094404
## 18 2017     6 4.574833  0.074833218  4.503284  0.003284395  4.483601
##       VAR12.err
## 1   0.238999864
## 2  -0.096163848
## 3  -0.187048880
## 4  -0.192955419
## 5   0.196481988
## 6  -0.307106409
## 7  -0.037576563
## 8  -0.201289249
## 9  -0.035206560
## 10 -0.041226819
## 11  0.305915000
## 12 -0.072028593
## 13  0.241262228
## 14  0.072460961
## 15  0.034661522
## 16 -0.030972640
## 17 -0.005595562
## 18 -0.016398607
```

VAR(12) RMSE: 0.1634807

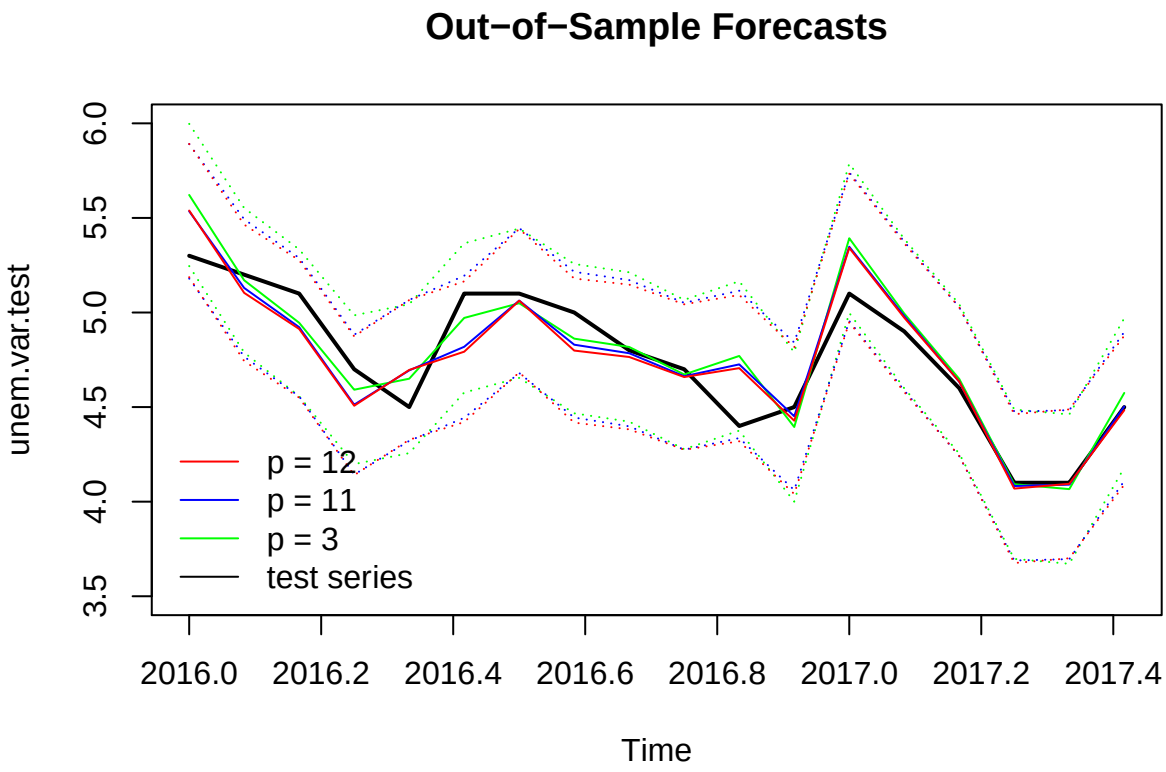
VAR(11) RMSE: 0.1587485

VAR(3) RMSE: 0.1587414

In terms of out-of-sample comparison, the VAR(3) performs marginally better than the other two. This is a small contradiction with the residual plot studies and serial correlation test. Instead, the RMSE results align with the preferred lag order estimated by BIC. Recall that the RMSE for our SARIMA(2,1,1)(1,0,1) is 0.1472144, so the SARIMA model is still better in terms of out-of-sample performance.

The three VAR models perform very closely, as illustrated by the forecast plot below. The VAR(3) forecast appears more sensitive to the observations at June 2016, which is probably the reason that its RMSE is lower than the other two models. Notice that the confidence bound of all three VAR models are narrower and more consistent than the SARIMA(2,1,1)(1,0,1) model.

```
plot(unem.var.test, ylim = c(3.5, 6), main = "Out-of-Sample Forecasts",
    lwd = 2)

lines(y = f.var.3$var.fore, x = as.numeric(time(unem.var.test)),
    col = "green")
lines(y = f.var.11$var.fore, x = as.numeric(time(unem.var.test)),
    col = "blue")
lines(y = f.var.12$var.fore, x = as.numeric(time(unem.var.test)),
    col = "red")

lines(y = f.var.3$var.low.ci, x = as.numeric(time(unem.var.test)),
    col = "green", lty = "dotted")
lines(y = f.var.3$var.upp.ci, x = as.numeric(time(unem.var.test)),
    col = "green", lty = "dotted")
lines(y = f.var.11$var.low.ci, x = as.numeric(time(unem.var.test)),
    col = "blue", lty = "dotted")
lines(y = f.var.11$var.upp.ci, x = as.numeric(time(unem.var.test)),
    col = "blue", lty = "dotted")
lines(y = f.var.12$var.low.ci, x = as.numeric(time(unem.var.test)),
    col = "red", lty = "dotted")
lines(y = f.var.12$var.upp.ci, x = as.numeric(time(unem.var.test)),
    col = "red", lty = "dotted")

legend("bottomleft", legend = c("p = 12", "p = 11", "p = 3",
    "test series"), lty = c("solid", "solid", "solid", "solid"),
    bty = "n", col = c("red", "blue", "green", "black"))
```



Final Model

Given that the residuals of the VAR(3) model have very mild serial correlation, marginally better RMSE, and much lower lag order than the VAR(11) and VAR(12) models, we should strongly consider it as the final choice. In the following, we examine the significance and interpretation of its coefficients.

```
summary(var.diff.mod.lag3)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: unem.var.train.diff, auto.var.train.diff
## Deterministic variables: const
## Sample size: 476
## Log Likelihood: -2730.571
## Roots of the characteristic polynomial:
## 0.6688 0.6688 0.6649 0.5593 0.5017 0.5017
## Call:
## VAR(y = cbind(unem.var.train.diff, auto.var.train.diff), p = 3,
##     season = 12L)
##
##
## Estimation results for equation unem.var.train.diff:
## ====================================================
## unem.var.train.diff = unem.var.train.diff.l1 + auto.var.train.diff.l1 + unem.var.train.diff.l2 + auto
##
##                           Estimate Std. Error t value Pr(>|t|)
## unem.var.train.diff.l1 -2.955e-03  4.599e-02  -0.064 0.948787
## auto.var.train.diff.l1 -3.424e-04  8.867e-05  -3.862 0.000129 ***
```

```
## unem.var.train.diff.l2  1.641e-01  4.509e-02   3.639 0.000305 ***
## auto.var.train.diff.l2 -2.699e-04  9.653e-05  -2.796 0.005387 **
## unem.var.train.diff.l3  1.591e-01  4.534e-02   3.510 0.000493 ***
## auto.var.train.diff.l3 -1.394e-04  8.986e-05  -1.552 0.121400
## const                  -2.672e-03  8.803e-03  -0.303 0.761647
## sd1                    -1.075e+00  6.544e-02 -16.422  < 2e-16 ***
## sd2                    -1.304e+00  5.878e-02 -22.184  < 2e-16 ***
## sd3                    -1.455e+00  7.243e-02 -20.094  < 2e-16 ***
## sd4                    -8.404e-01  5.847e-02 -14.373  < 2e-16 ***
## sd5                    -2.696e-01  5.344e-02  -5.045 6.54e-07 ***
## sd6                    -8.708e-01  5.601e-02 -15.548  < 2e-16 ***
## sd7                    -1.240e+00  4.894e-02 -25.331  < 2e-16 ***
## sd8                    -1.176e+00  5.438e-02 -21.621  < 2e-16 ***
## sd9                    -1.034e+00  5.022e-02 -20.580  < 2e-16 ***
## sd10                   -7.967e-01  4.586e-02 -17.373  < 2e-16 ***
## sd11                   -9.107e-01  4.853e-02 -18.767  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.1918 on 458 degrees of freedom
## Multiple R-Squared: 0.779,   Adjusted R-squared: 0.7708
## F-statistic: 94.99 on 17 and 458 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation auto.var.train.diff:
## ====================================================
## auto.var.train.diff = unem.var.train.diff.l1 + auto.var.train.diff.l1 + unem.var.train.diff.l2 + aut
##
##                          Estimate Std. Error t value Pr(>|t|)
## unem.var.train.diff.l1 -59.89492   23.58542  -2.539 0.011431 *
## auto.var.train.diff.l1  -0.61689    0.04548 -13.564  < 2e-16 ***
## unem.var.train.diff.l2 -20.10760   23.12561  -0.869 0.385032
## auto.var.train.diff.l2  -0.48635    0.04951  -9.824  < 2e-16 ***
## unem.var.train.diff.l3 -25.04746   23.25502  -1.077 0.282012
## auto.var.train.diff.l3  -0.24112    0.04609  -5.232 2.56e-07 ***
## const                    1.68305    4.51490   0.373 0.709487
## sd1                    254.58641   33.56128   7.586 1.86e-13 ***
## sd2                    429.29052   30.14939  14.239  < 2e-16 ***
## sd3                    238.22932   37.14846   6.413 3.56e-10 ***
## sd4                    323.25969   29.98816  10.780  < 2e-16 ***
## sd5                    194.73408   27.40927   7.105 4.64e-12 ***
## sd6                    119.09833   28.72532   4.146 4.03e-05 ***
## sd7                    160.63555   25.10271   6.399 3.87e-10 ***
## sd8                     27.27200   27.88862   0.978 0.328645
## sd9                     89.38955   25.75781   3.470 0.000569 ***
## sd10                    17.67736   23.52082   0.752 0.452700
## sd11                   184.94625   24.88768   7.431 5.32e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 98.36 on 458 degrees of freedom
## Multiple R-Squared: 0.6542,  Adjusted R-squared: 0.6414
```

```
## F-statistic: 50.97 on 17 and 458 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##                      unem.var.train.diff auto.var.train.diff
## unem.var.train.diff             0.03678             -0.3442
## auto.var.train.diff            -0.34415           9675.6369
##
## Correlation matrix of residuals:
##                      unem.var.train.diff auto.var.train.diff
## unem.var.train.diff             1.00000            -0.01824
## auto.var.train.diff            -0.01824             1.00000
```

From the model summary, we observe:

1) Coefficients of auto sales lags are generally significant and negative. This confirms our intuition in the EDA that auto sales leads unemployment rate in opposite directions.
2) Coefficients of unemployment lags on unemployment rate are close to zero or generally positive, which is somewhat intuitive. If unemployment rate was high 2-3 months ago, it will probably stay high, especially in times of persistent economic boom or busts.
3) Coefficients of unemployment lags are not significant on auto sales. This confirms our EDA insight that auto sales leads unemployment, not the other way around.

The estimated VAR(3) model specification is as follows:

$$\begin{aligned}\hat{x_t} = &-0.002672 - 0.002955 \cdot x_{t-1} - 0.0003424 \cdot y_{t-1} + 0.1641 \cdot x_{t-2} - 0.0002699 \cdot y_{t-2} \\ &+0.1591 \cdot x_{t-3} - 0.0001394 \cdot y_{t-3} - 1.075 \cdot sd1 - 1.304 \cdot sd2 - 1.455 \cdot sd3 - 0.8404 \cdot sd4 \\ &-0.2696 \cdot sd5 - 0.8708 \cdot sd6 - 1.24 \cdot sd7 - 1.176 \cdot sd8 - 1.034 \cdot sd9 - 0.7967 \cdot sd10 \\ &-0.9107 \cdot sd11\end{aligned}$$

$$\begin{aligned}\hat{y_t} = &1.68305 - 59.89492 \cdot x_{t-1} - 0.61689 \cdot y_{t-1} - 20.10760 \cdot x_{t-2} - 0.48635 \cdot y_{t-2} \\ &-25.04746 \cdot x_{t-3} - 0.24112 \cdot y_{t-3} + 254.58641 \cdot sd1 + 429.29052 \cdot sd2 + 238.22932 \cdot sd3 \\ &+323.25969 \cdot sd4 + 194.73408 \cdot sd5 + 119.09833 \cdot sd6 + 160.63555 \cdot sd7 + 27.27200 \cdot sd8 \\ &+89.38955 \cdot sd9 + 17.67736 \cdot sd10 + 184.94625 \cdot sd11\end{aligned}$$

where the two respective residual series are bivariate white noises.

To check for stationarity, the characteristic function can be evaluated using the determinant:

$$\begin{aligned}\hat{x_t} = &-0.002672 - 0.002955 \cdot x_{t-1} - 0.0003424 \cdot y_{t-1} + 0.1641 \cdot x_{t-2} - 0.0002699 \cdot y_{t-2} \\ &+0.1591 \cdot x_{t-3} - 0.0001394 \cdot y_{t-3} - 1.075 \cdot sd1 - 1.304 \cdot sd2 - 1.455 \cdot sd3 - 0.8404 \cdot sd4 \\ &-0.2696 \cdot sd5 - 0.8708 \cdot sd6 - 1.24 \cdot sd7 - 1.176 \cdot sd8 - 1.034 \cdot sd9 - 0.7967 \cdot sd10 \\ &-0.9107 \cdot sd11\end{aligned}$$

$$\begin{aligned}\hat{y_t} = 1.68305 - 59.89492 \cdot x_{t-1} - 0.61689 \cdot y_{t-1} - 20.10760 \cdot x_{t-2} - 0.48635 \cdot y_{t-2} \\ -25.04746 \cdot x_{t-3} - 0.24112 \cdot y_{t-3} + 254.58641 \cdot sd1 + 429.29052 \cdot sd2 + 238.22932 \cdot sd3 \\ +323.25969 \cdot sd4 + 194.73408 \cdot sd5 + 119.09833 \cdot sd6 + 160.63555 \cdot sd7 + 27.27200 \cdot sd8 \\ +89.38955 \cdot sd9 + 17.67736 \cdot sd10 + 184.94625 \cdot sd11\end{aligned}$$

where the two respective residual series are bivariate white noises.

To check for stationarity, the characteristic function can be evaluated using the determinant:

$$\begin{aligned}&\left|\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} - \begin{pmatrix}-0.002955 & -0.0003424\\ -59.89492 & -0.61689\end{pmatrix} B - \begin{pmatrix}0.1641 & -0.0002699\\ -20.10760 & -0.48635\end{pmatrix} B^2 - \begin{pmatrix}0.1591 & -0.0001394\\ -25.04746 & -0.24112\end{pmatrix} B^3\right| \\ &= \begin{vmatrix}(1 + 0.002955B - 0.1641B^2 - 0.1591B^3) & (0.0003424B + 0.0002699B^2 + 0.0001394B^3)\\ (59.89492B + 20.10760B^2 + 25.04746B^3) & (1 + 0.61689B + 0.48635B^2 + 0.24112B^3)\end{vmatrix} \\ &= -0.04185381B^6 - 0.1265094B^5 - 0.1995974B^4 - 0.04082497B^3 + 0.3035649B^2 + 0.6198450000000001B + 1\end{aligned}$$

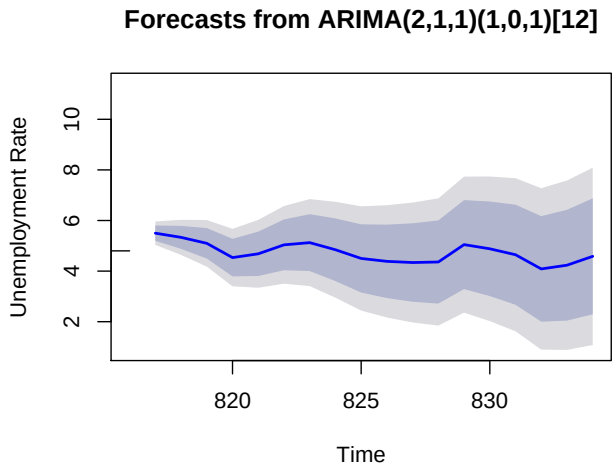
From this it can be verified that the fitted VAR(3) model is stationary since all the roots exceed unity in absolute value:

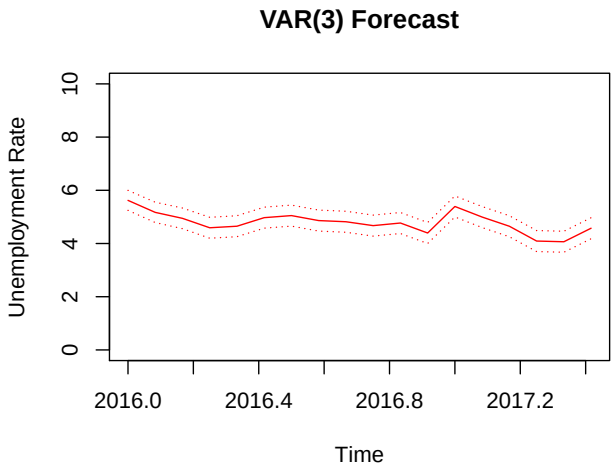
```
Mod(polyroot(c(1, 0.619845, 0.3035649, -0.04082497, -0.1995974,
    -0.1265094, -0.04185381)))
```

```
## [1] 1.495320 1.787943 1.495320 1.504115 1.993337 1.993337
```

Comparing the VAR(3) Model Against the SARIMA(2,1,1)(1,0,1)[12] Model:

```
plot(forecast(m.diff.3, h = 18), xlim = c(816, 834), xlab = "Time",
    ylab = "Unemployment Rate")

plot(y = f.var.3$var.fore, x = as.numeric(time(unem.var.test)),
    type = "l", col = "red", xlab = "Time", ylab = "Unemployment Rate",
    main = "VAR(3) Forecast", ylim = c(0, 10))
lines(y = f.var.3$var.low.ci, x = as.numeric(time(unem.var.test)),
    col = "red", lty = "dotted")
lines(y = f.var.3$var.upp.ci, x = as.numeric(time(unem.var.test)),
    col = "red", lty = "dotted")
```





In terms of in-sample fit, both models approximate the raw series well, as depicted by the earlier time plots. In terms of out-of-sample 18 steps ahead performance, both models had close RMSE measures. The two models are so similar in these accuracy measures possibly because 1) Both VAR and SARIMA models perform step by step forecasts that take advantage of autocorrelations with the series's own lags, with heavy reliance on the past observations in the raw series. This is very different from the linear regression model, which can only predict with the time indexes, so information is much more restricted. 2) Both the VAR(3) and SARIMA(2,1,1)(1,0,1)[12] models are integrated by order 1 (first differenced series as input) and incorporate AR components of similar order.

However, variance of their forecasts are very different. Variance of the SARIMA model grows drastically with time because its step-by-step forecast has to depend more and more on estimated instead of observed lag values as time goes on. Each estimated lag value contributes its own uncertainty towards the next forecast. On the other hand, although the VAR model has a similar forecast mechanism, it has much more consistent forecast variance. By accounting for cross-correlation in the raw series and thus modeling residuals as bivariate white noise, the VAR mechanism manages to impose heavier structure on the residuals (thus coefficient estimates and forecasts) so they seem more restricted than the SARIMA model.

One may be surpised that the VAR(3) model does not perform much better than the SARIMA(2,1,1)(1,0,1)[12] model, given that we have useful information of auto sales that leads unemployment rate. This is because: 1) The VAR model limits us to auto-regressive terms, whereas the SARIMA model allows moving average terms to directly account for moving average components. 2) The VAR model can only account for seasonal patterns with indicator variables; we are deprived of the option to account for seasonal auto-regressive or seasonal moving average terms.

In conclusion, to forecast unemployment rate, one can consider a SARIMA model for more unbiased estimators, or a VAR model for more precise predictions.