Project

Mahesh Pandit

11/21/2020

Read the data and transform it into a time series

```r
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

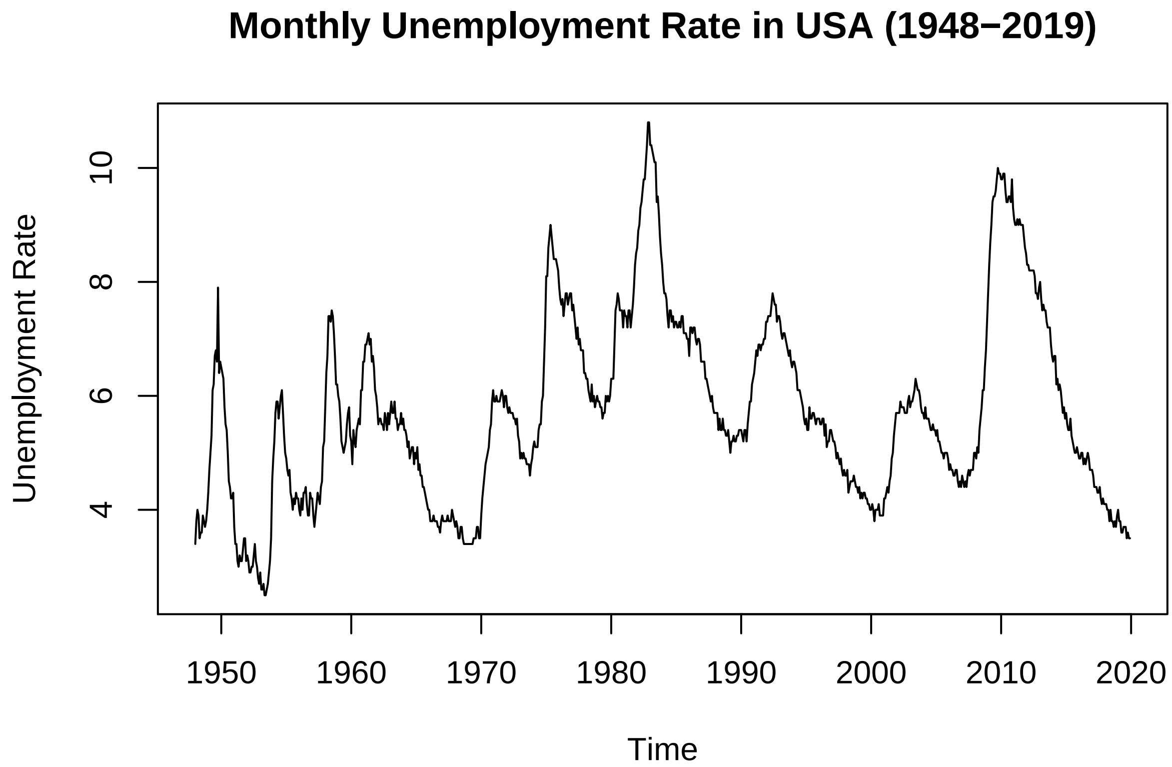
```r
data = read.csv("USUnemployment.csv", header = T)

head(data)
```

```
##   Year Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 1 1948 3.4 3.8 4.0 3.9 3.5 3.6 3.6 3.9 3.8 3.7 3.8 4.0
## 2 1949 4.3 4.7 5.0 5.3 6.1 6.2 6.7 6.8 6.6 7.9 6.4 6.6
## 3 1950 6.5 6.4 6.3 5.8 5.5 5.4 5.0 4.5 4.4 4.2 4.2 4.3
## 4 1951 3.7 3.4 3.4 3.1 3.0 3.2 3.1 3.1 3.3 3.5 3.5 3.1
## 5 1952 3.2 3.1 2.9 2.9 3.0 3.0 3.2 3.4 3.1 3.0 2.8 2.7
## 6 1953 2.9 2.6 2.6 2.7 2.5 2.5 2.6 2.7 2.9 3.1 3.5 4.5
```

```r
tail(data)
```

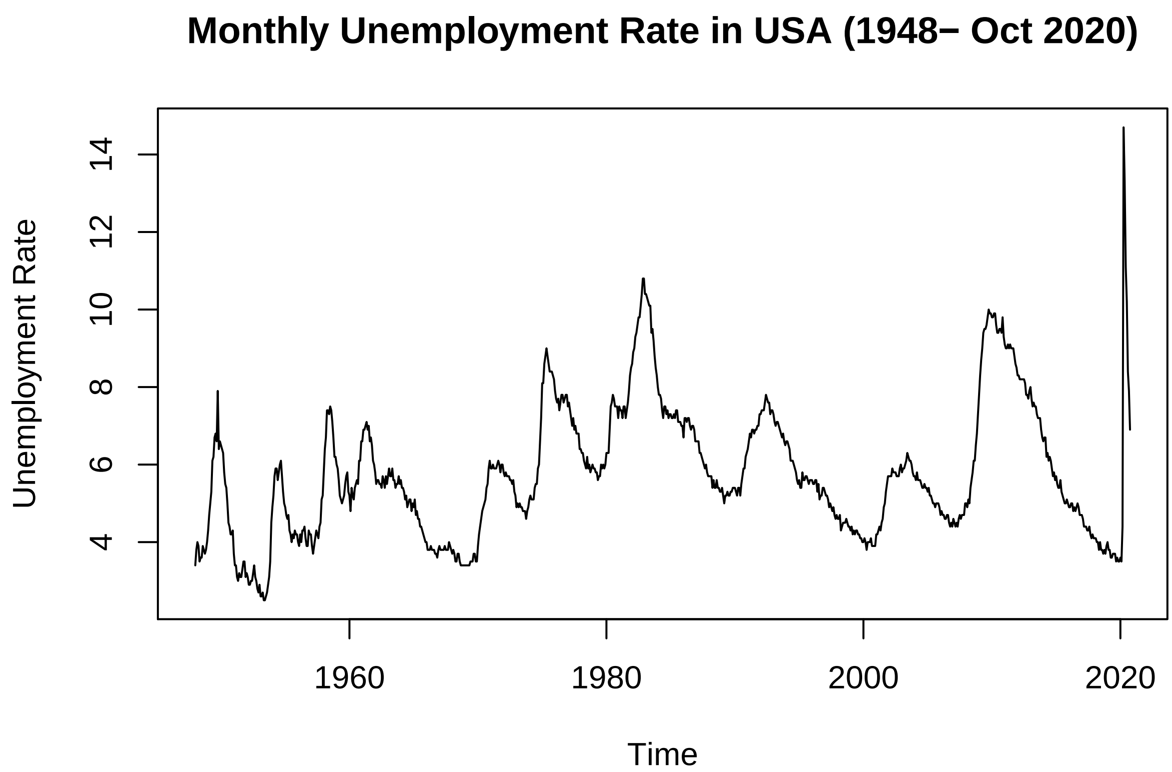
```
##    Year Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 67 2014 6.6 6.7 6.7 6.2 6.3 6.1 6.2 6.1 5.9 5.7 5.8 5.6
## 68 2015 5.7 5.5 5.4 5.4 5.6 5.3 5.2 5.1 5.0 5.0 5.1 5.0
## 69 2016 4.9 4.9 5.0 5.0 4.8 4.9 4.8 4.9 5.0 4.9 4.7 4.7
## 70 2017 4.7 4.6 4.4 4.4 4.4 4.3 4.3 4.4 4.2 4.1 4.2 4.1
## 71 2018 4.1 4.1 4.0 4.0 3.8 4.0 3.8 3.8 3.7 3.8 3.7 3.9
## 72 2019 4.0 3.8 3.8 3.6 3.6 3.7 3.7 3.7 3.5 3.6 3.5 3.5
```

```r
data.ts = ts(c(unname(t(data))[2:13,]), start = c(1948, 1), end = c(2019, 12), frequency = 12 )

plot(data.ts, main = "Monthly Unemployment Rate in USA (1948-2019)", ylab = "Unemployment Rate")
```



Include 2020 data

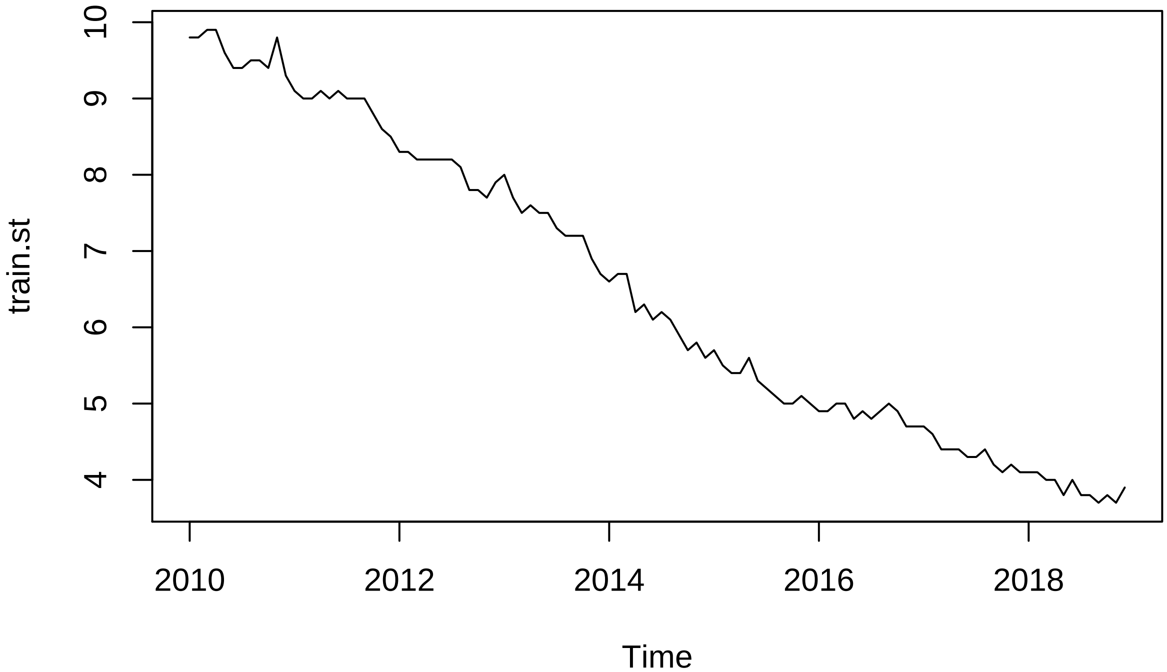
```
data_2020 = readxl::read_excel('USUnemployment_2020.xlsx')
data.2020.ts = ts(data_2020$`Unemployment Rate`, start = c(2020, 1), frequency = 12)

data.combined = ts(c(data.ts, data.2020.ts), start = start(data.ts), frequency = 12)
plot(data.combined, main = "Monthly Unemployment Rate in USA (1948- Oct 2020)", ylab = "Unemployment Ra
```



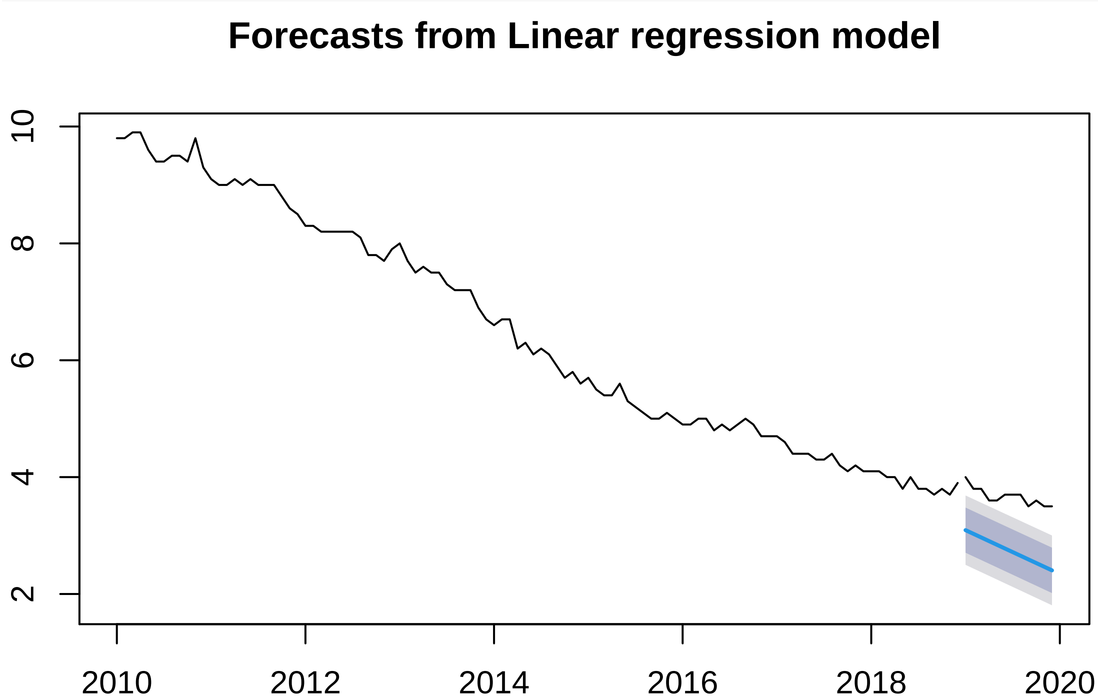
The data seems to have a cyclic pattern with 10-15 year cycles

Perform short term forecasts for 2020 using data from the past decade.

```
library(forecast)

train.st = window(data.ts, start = c(2010, 1), end = c(2018, 12))
valid.st = window(data.ts, start = c(2019, 1))

plot(train.st)
```




```
#Use a linear model with trend
m1.st = tslm( train.st ~ trend )
m1.st.forecast = forecast(m1.st, h = 12)

plot(m1.st.forecast)
lines(valid.st)
```

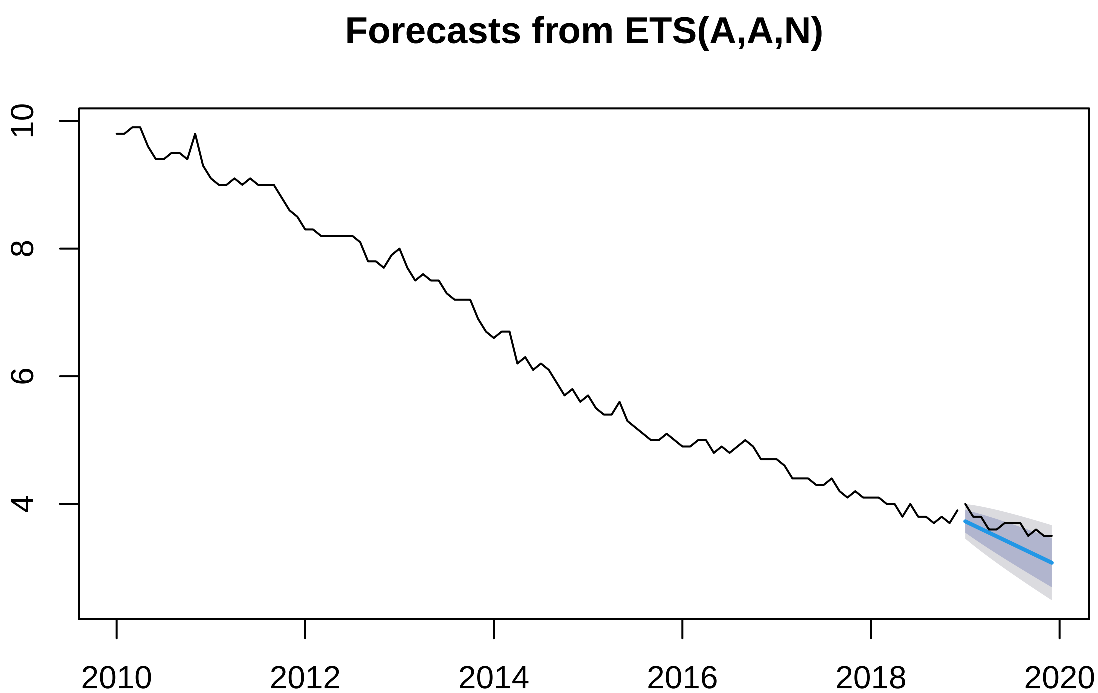



```
accuracy(m1.st.forecast, valid.st)
```

```
##                        ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 3.705512e-17 0.2911756 0.2304679 0.02219919 4.202131 0.3225206
```

```
## Test set     9.184517e-01 0.9271773 0.9184517 25.12349721 25.123497 1.2852968
##                   ACF1 Theil's U
## Training set 0.8510261        NA
## Test set     0.5806379  8.220472
```

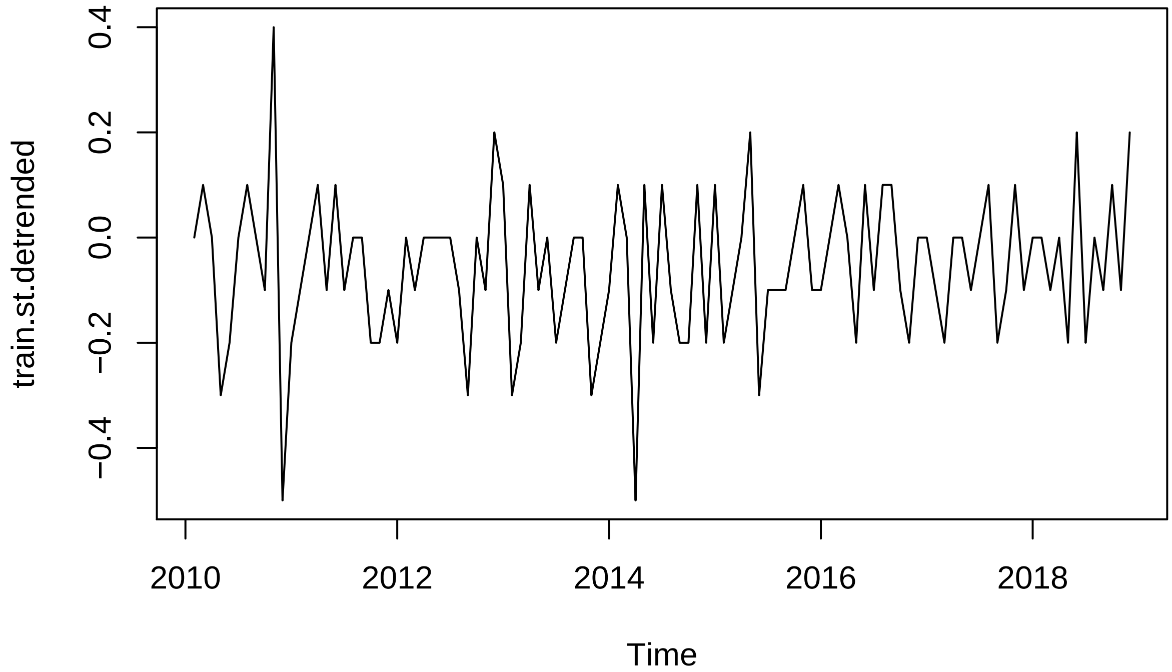
```r
#ETS model
m2.st = ets( train.st, model = 'ZZZ')
m2.st.forecast = forecast(m2.st, h = 12)

plot(m2.st.forecast)
lines(valid.st)
```



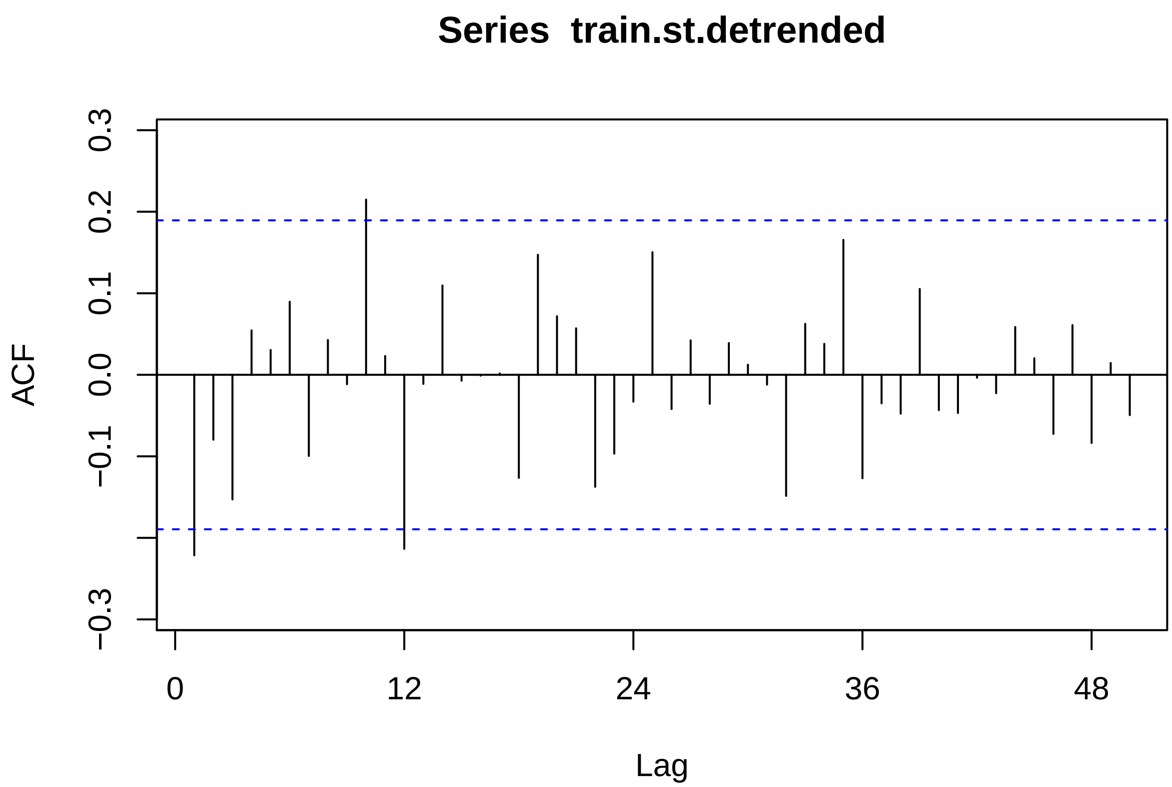

```r
accuracy(m2.st.forecast, valid.st)
```

```
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## Training set 0.00332433 0.1362710 0.1071784 0.1403971 1.723514 0.1499872
## Test set     0.26312509 0.2881598 0.2631251 7.2134476 7.213448 0.3682217
##                   ACF1 Theil's U
## Training set 0.1042555        NA
## Test set     0.5347437  2.590801
```

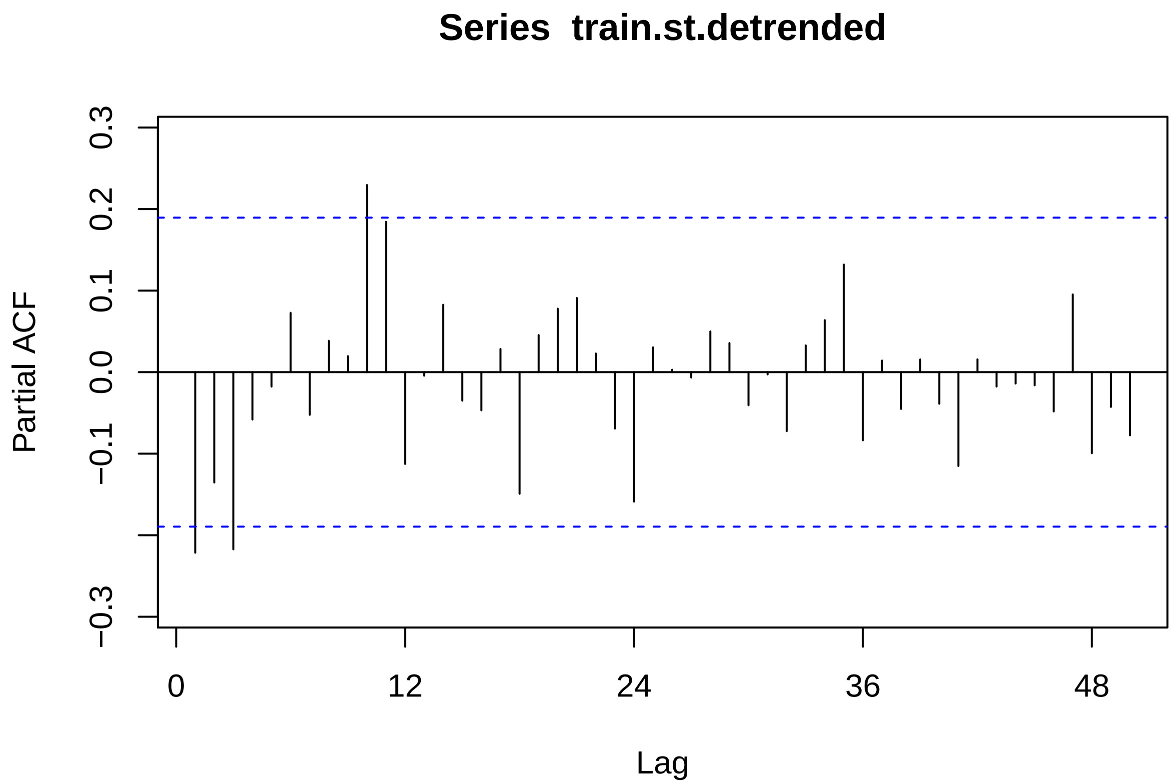
```r
#ARIMA model
train.st.detrended = diff(train.st, lag = 1)

plot(train.st.detrended)
```



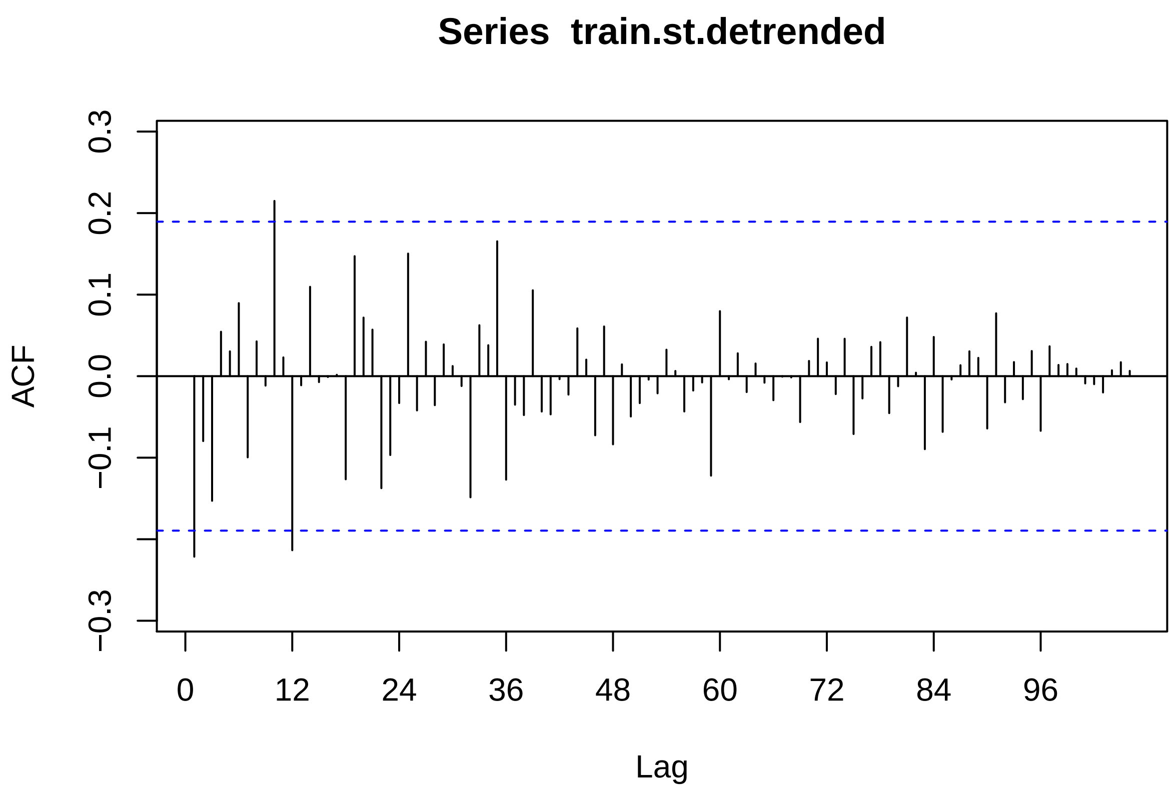

```
Acf(train.st.detrended, lag.max = 50)
```



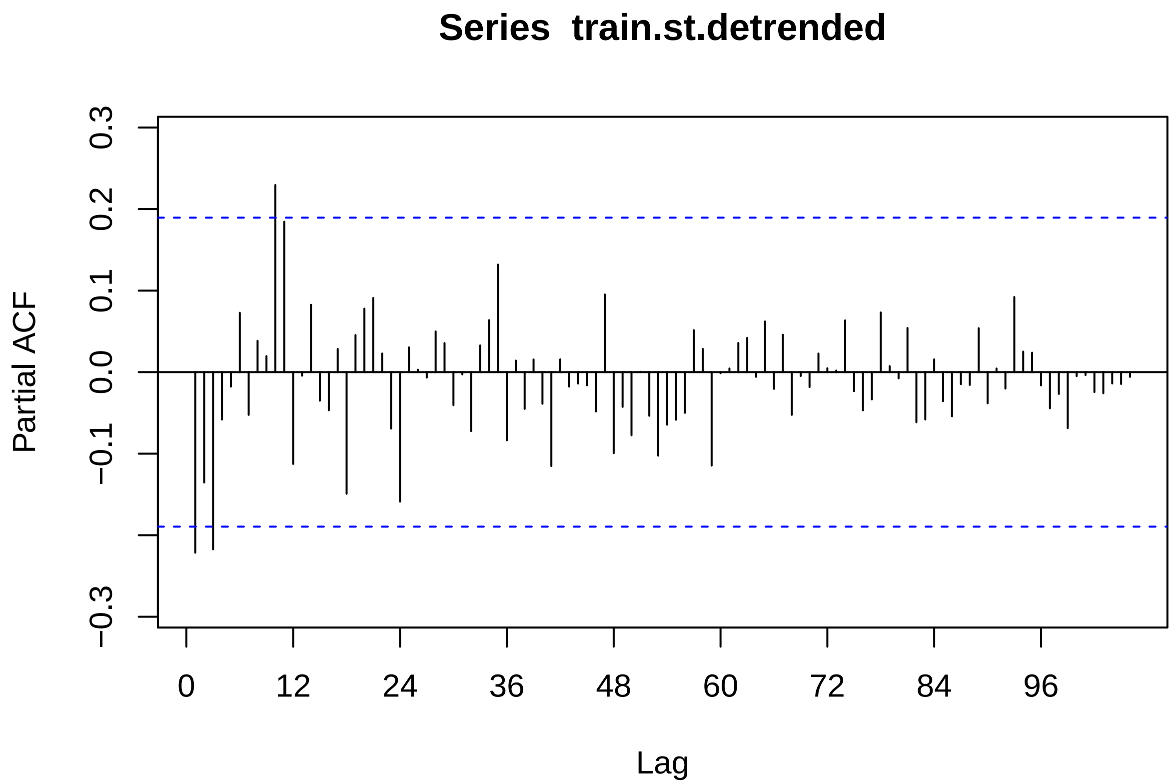

```
Pacf(train.st.detrended, lag.max = 50)
```



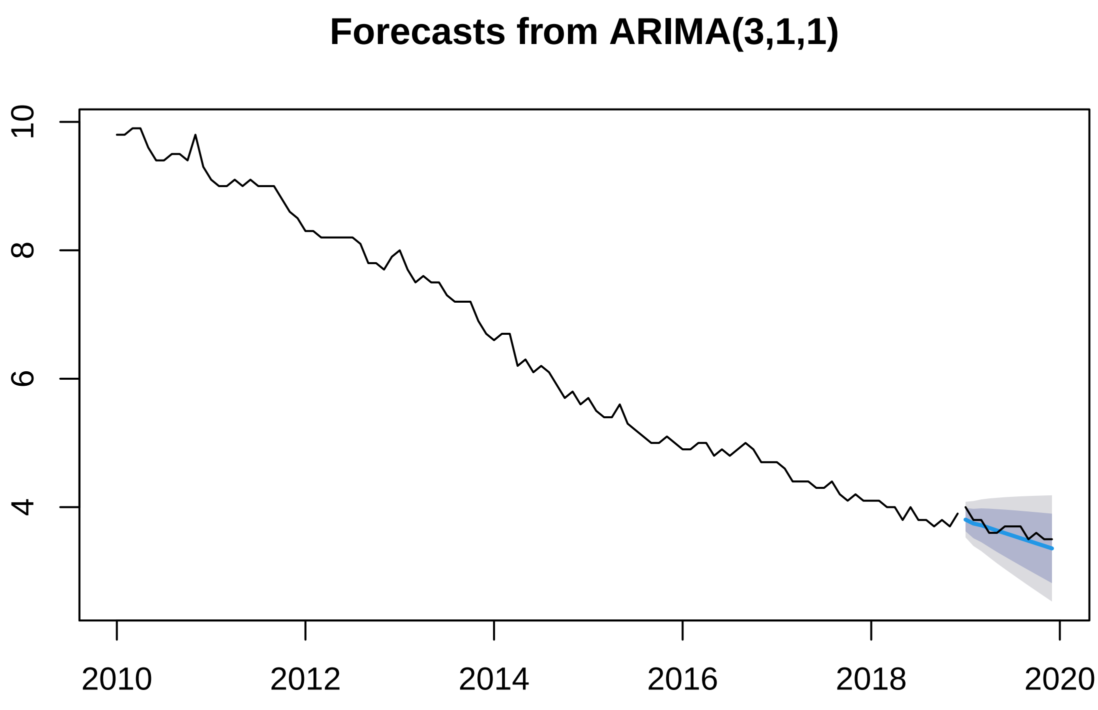

```r
Acf(train.st.detrended, lag.max = 150)
```




```r
Pacf(train.st.detrended, lag.max = 150)
```




```r
m3.st = Arima(train.st, order = c(3, 1, 1))
m3.st.forecast = forecast( m3.st, h = 12)

plot(m3.st.forecast)
lines(valid.st)
```

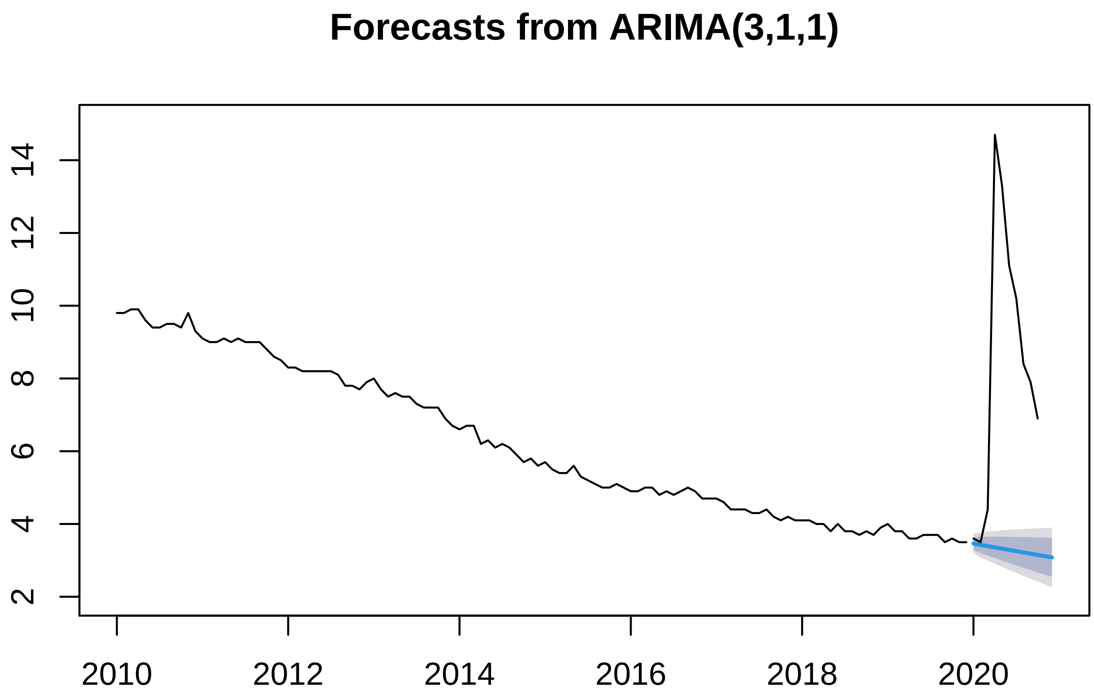


```r
accuracy(m3.st.forecast, valid.st)
```

```
##                        ME      RMSE       MAE         MPE     MAPE      MASE
## Training set -0.005457949 0.1382899 0.1075646 0.007652278 1.750418 0.1505278
## Test set      0.090412594 0.1220031 0.1091428 2.440494511 2.960778 0.1527363
##                     ACF1 Theil's U
## Training set -0.03265812        NA
## Test set      0.16826741  1.007049
```

The ARIMA model has the best accuracy and is most robust. Use it to perform short term forecast for 2020

```r
final.st = Arima(window(data.ts, start = c(2010, 1)), order = c(3, 1, 1))
final.st.forecast = forecast( final.st, h = 12 )

plot(final.st.forecast, ylim = c(2, 15))
lines(data.2020.ts)
```

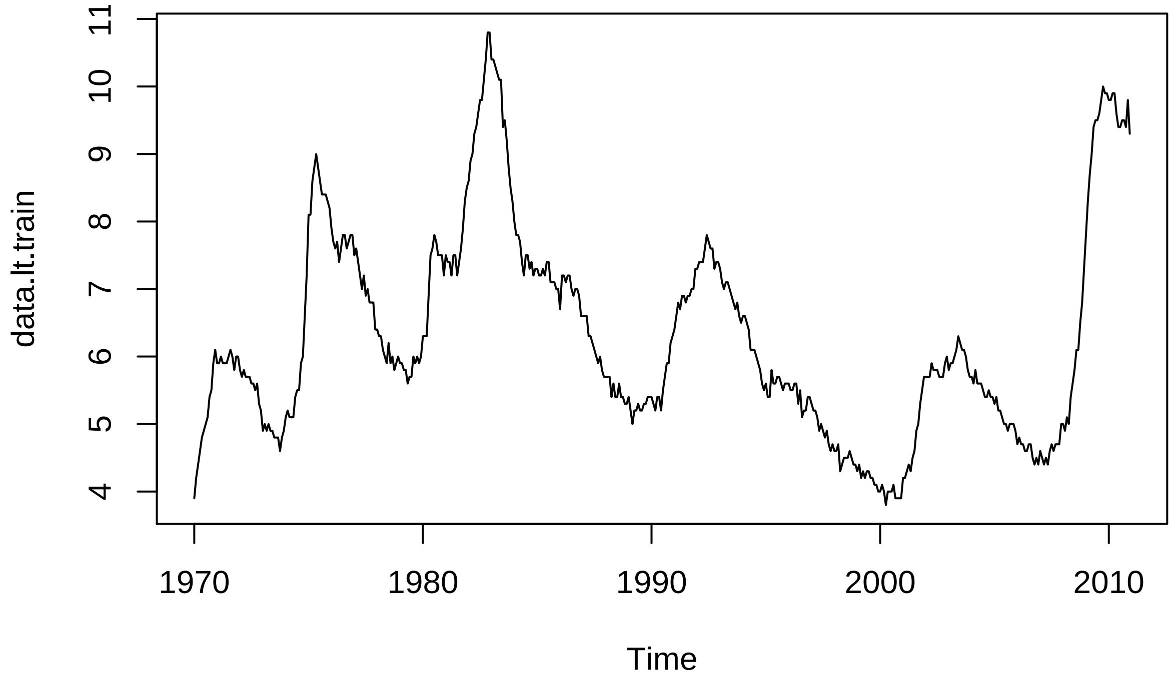


The short term forecast cannot predict the huge surge in unemployment in 2020

Perform long-term forecasts using data from 1970 onward

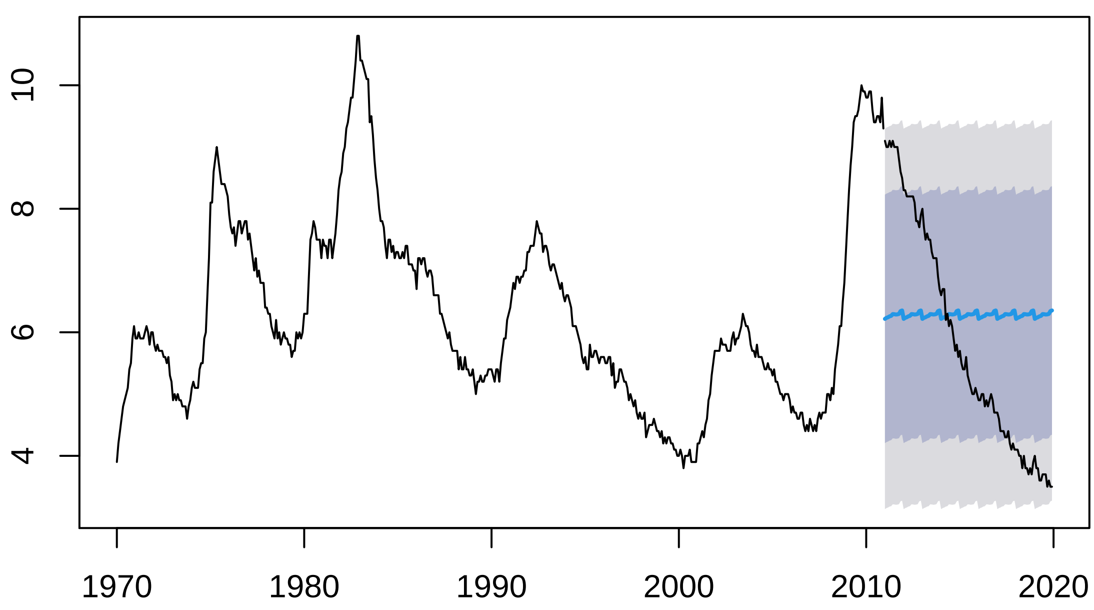
Use a simple linear model with seasonality

```r
data.lt = window(data.ts, start = c(1970, 1))
data.lt.train = window(data.lt, end = c(2010, 12))
data.lt.valid = window(data.lt, start = c(2011, 1))
plot(data.lt.train)
```




```r
m1.lt = tslm( data.lt.train ~ season )
m1.lt.forecast = forecast(m1.lt, h = length(data.lt.valid))
plot(m1.lt.forecast)
lines(data.lt.valid)
```

Forecasts from Linear regression model

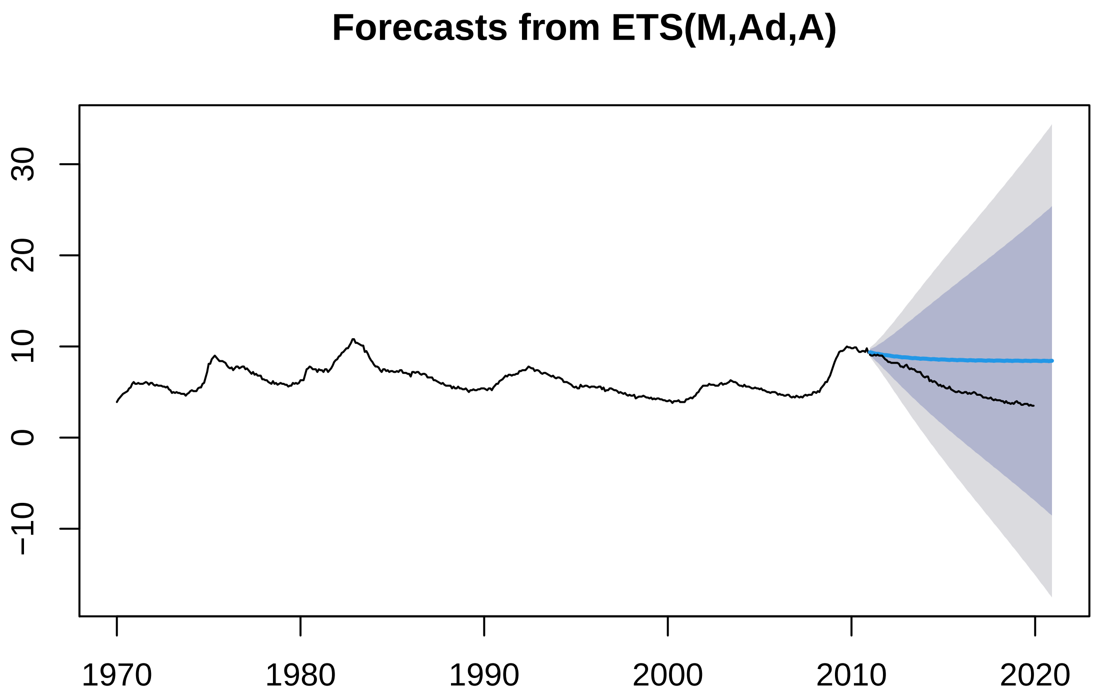


The simple linear model is unable to capture the cycles.

Try ETS model

```r
m2.lt = ets(data.lt.train, model = "MAA")
m2.lt.forecast = forecast(m2.lt, h = 120)
```

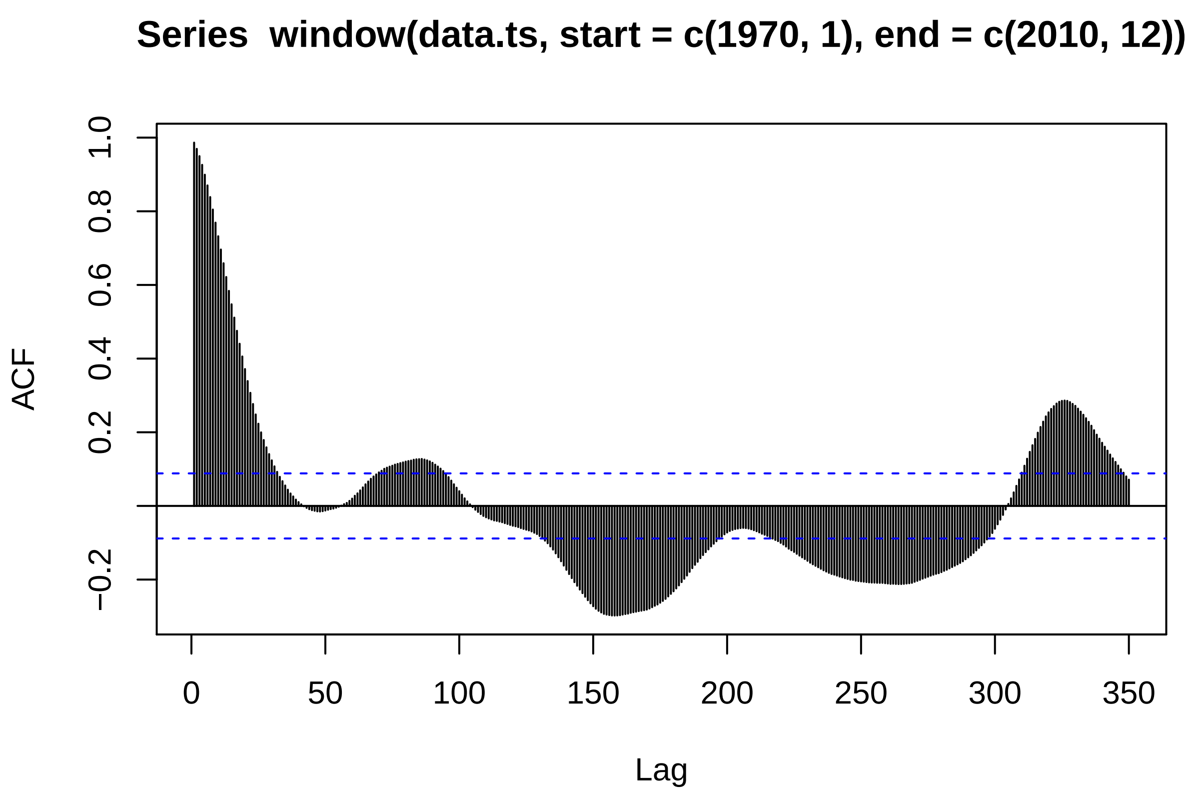
```r
plot(m2.lt.forecast)
lines(data.lt.valid)
```



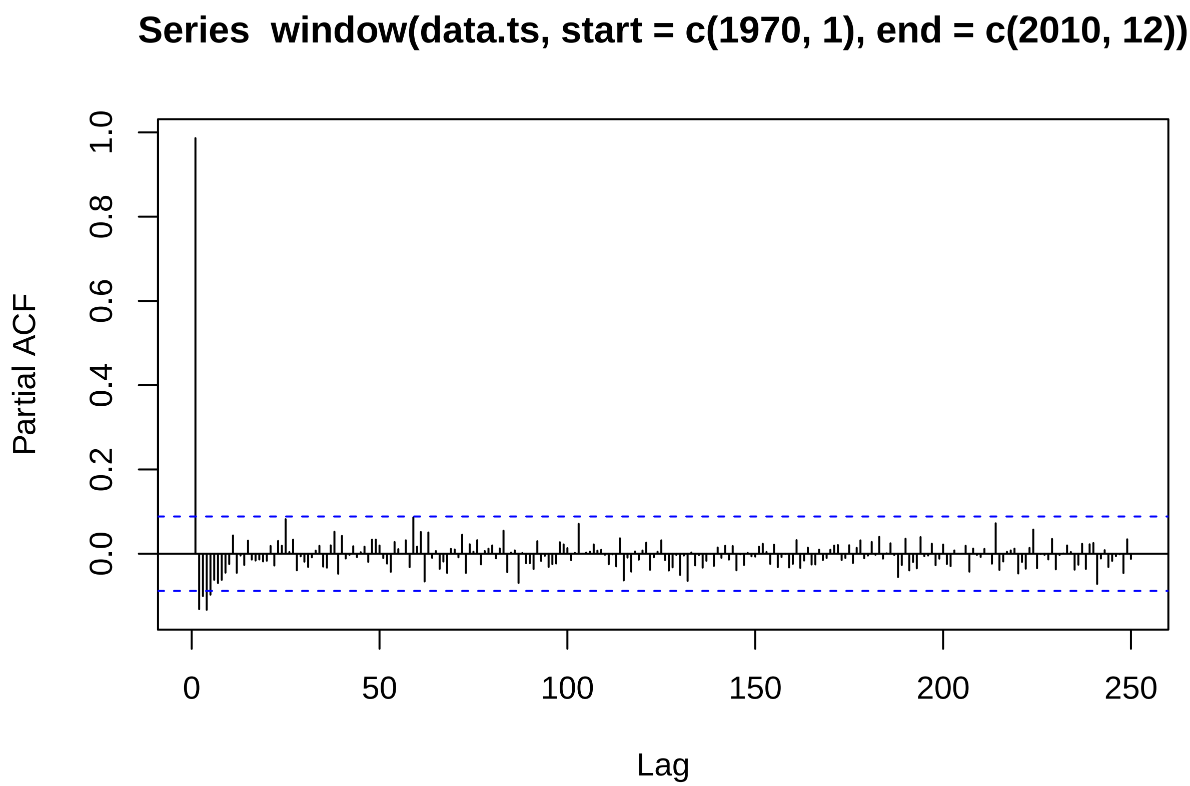
The ETS model is unable to capture the seasonality very well and the confidence intervals are extremely wide.

Try Arima models

```r
Acf(window(data.ts, start = c(1970, 1), end = c(2010, 12)), lag.max = 350)
```



```
Pacf(window(data.ts, start = c(1970, 1), end = c(2010, 12)), lag.max = 250)
```

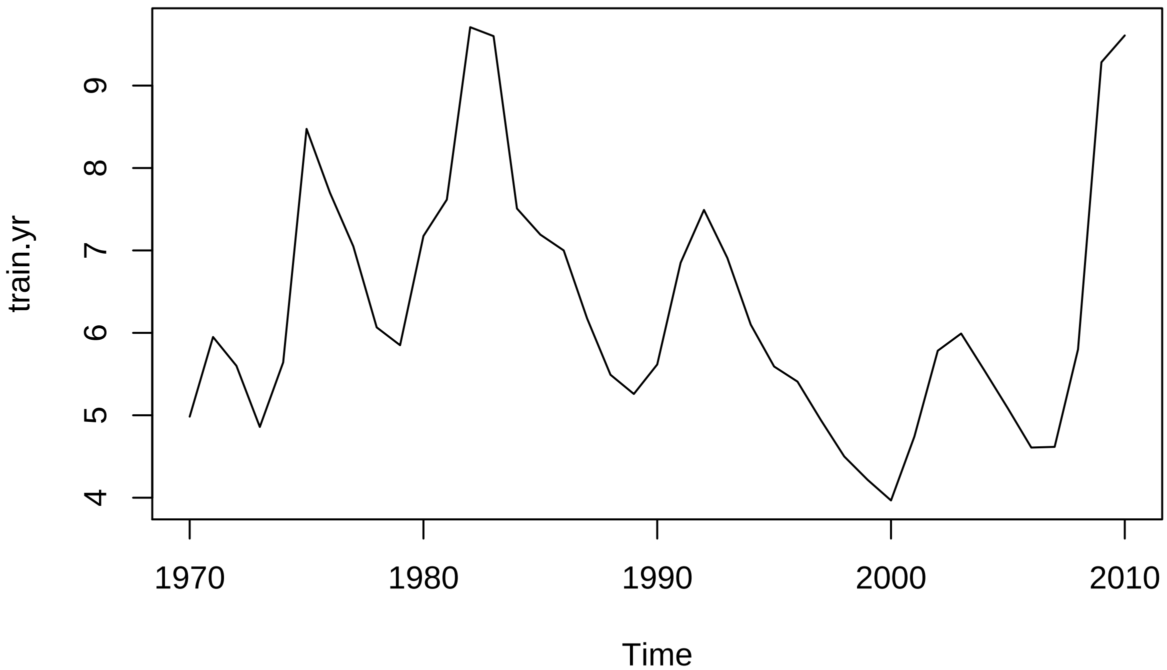



```
dt = window(data.ts, start = c(1970, 1), end = c(2010, 12))
# m = Arima(dt, order = c(2, 0, 0), seasonal = list(order = c(0, 0, 2), period = 156))
# m.forecast = forecast( m, h = 120)
# plot(m.forecast)
```

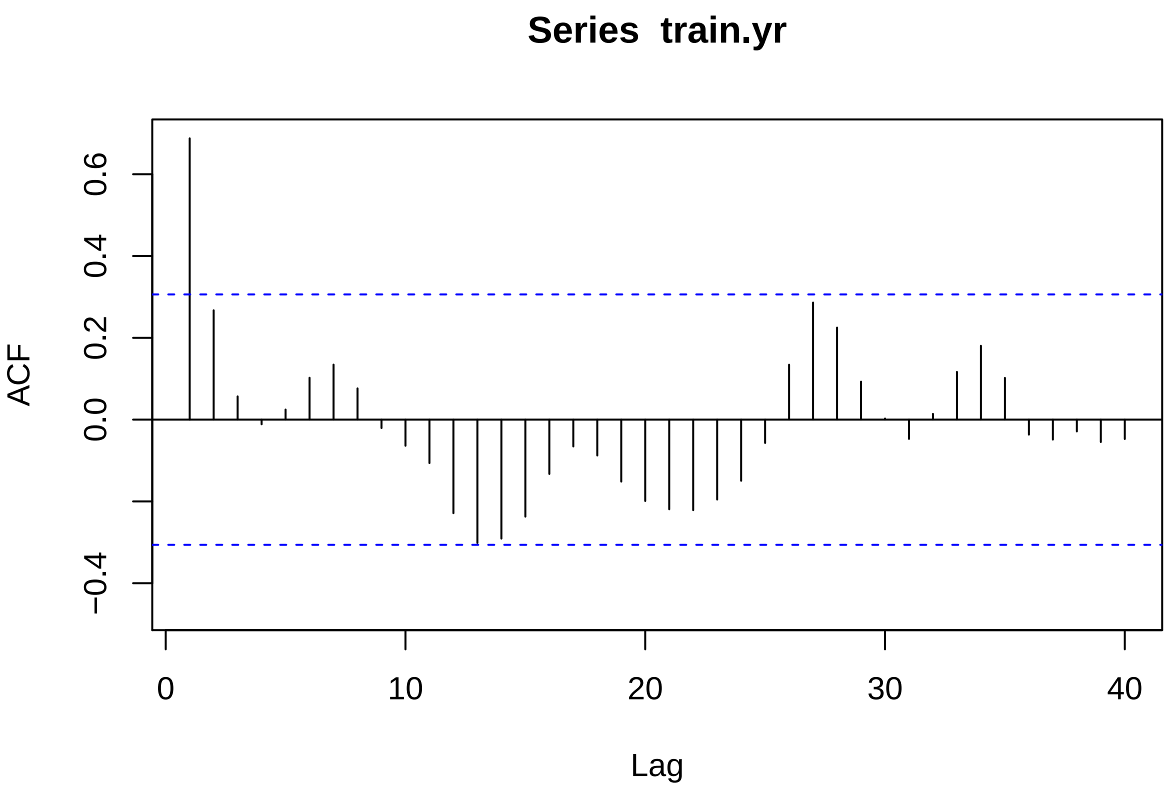
By looking at the ACF and PACF plots above, I tried running the ARIMA model above, but R could not compute it and timed out

I tried to solve this issue by aggregating the data to yearly averages.

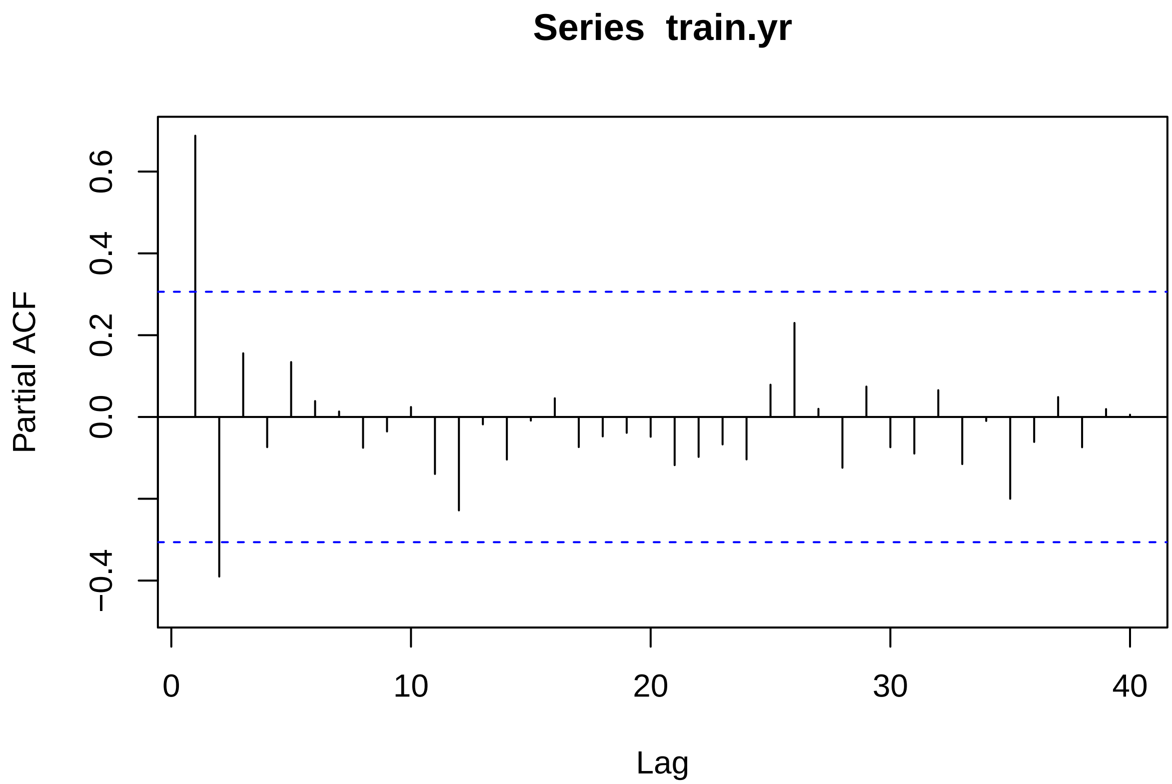
```
data.yr = aggregate(data.ts, FUN = mean)

train.yr = window(data.yr, start = 1970, end = 2010)
valid.yr = window(data.yr, start = 2011)

plot(train.yr)
```



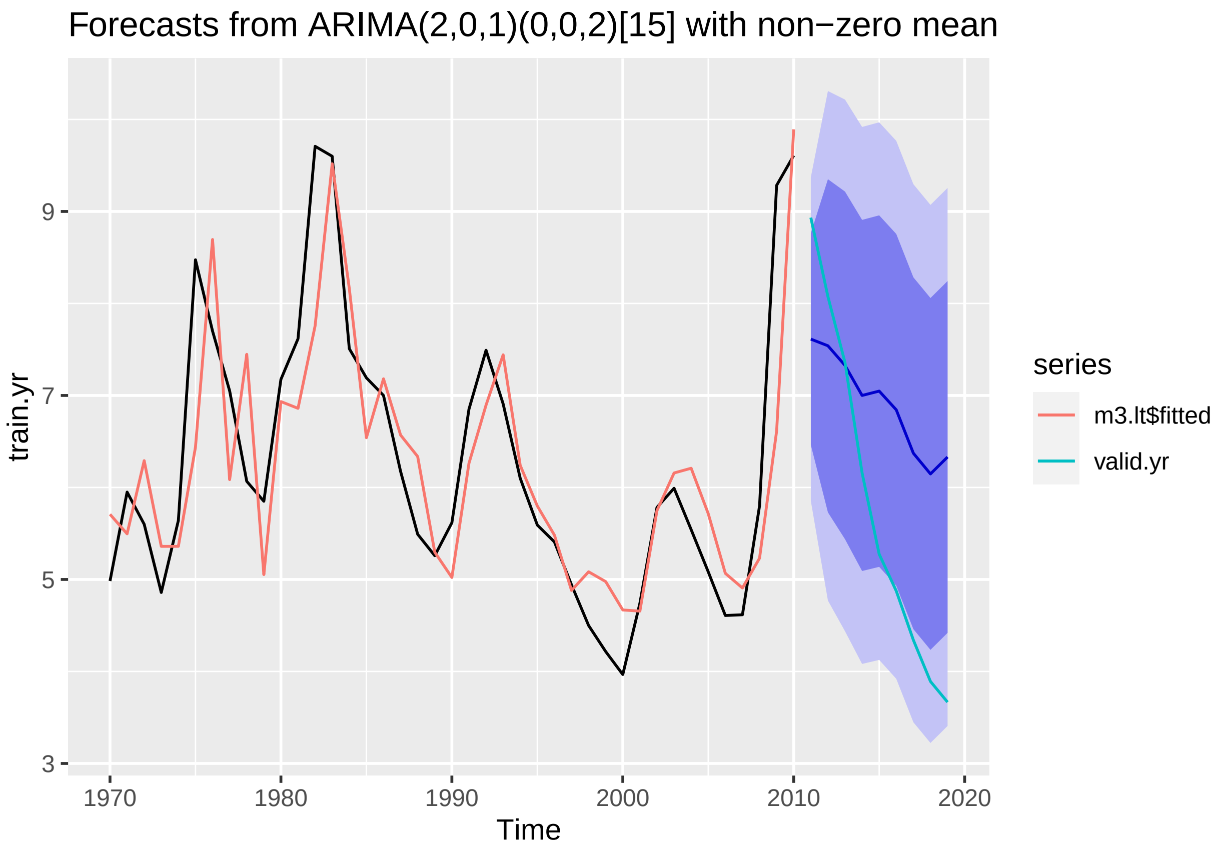

```r
Acf(train.yr, lag.max = 250)
```




```r
Pacf(train.yr, lag.max = 250)
```



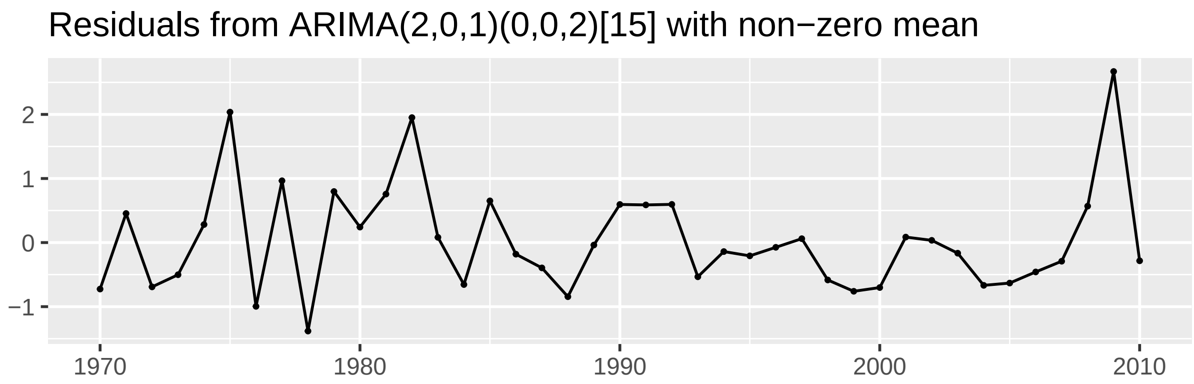

```r
m3.lt = Arima(train.yr, order = c(2, 0, 1), seasonal = list(order = c(0,0,2), period = 15) )
m3.lt.forecast = forecast( m3.lt, h = length(valid.yr))

autoplot(m3.lt.forecast)+autolayer(valid.yr)+autolayer(m3.lt$fitted)
```

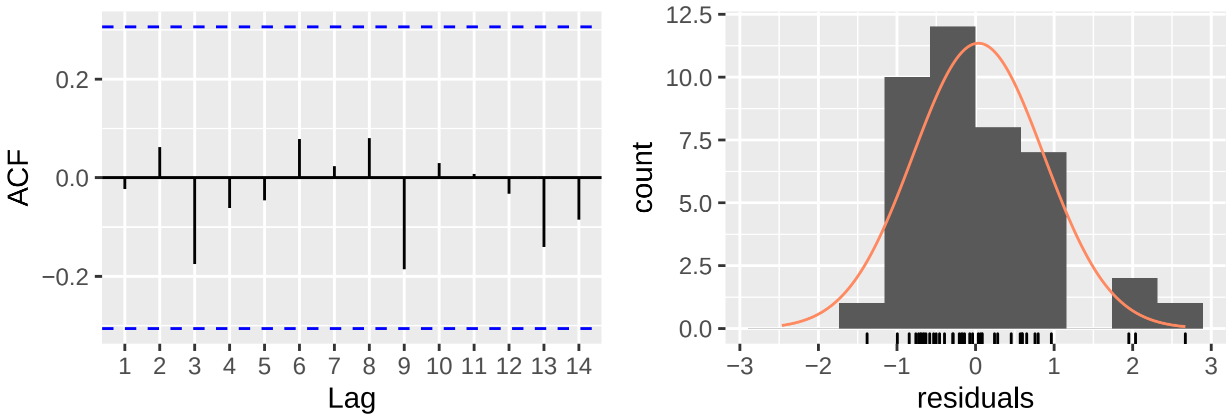



```r
accuracy( m3.lt.forecast, valid.yr )
```

```
##                        ME      RMSE      MAE         MPE      MAPE      MASE
## Training set  0.03675409 0.8247055 0.617644  -0.9844881  9.599456 0.8027867
## Test set     -1.07190728 1.7019422 1.491212 -27.0252937 31.877163 1.9382126
##                     ACF1 Theil's U
## Training set -0.02253484        NA
## Test set      0.64964942  3.461259
```

```r
checkresiduals(m3.lt.forecast)
```



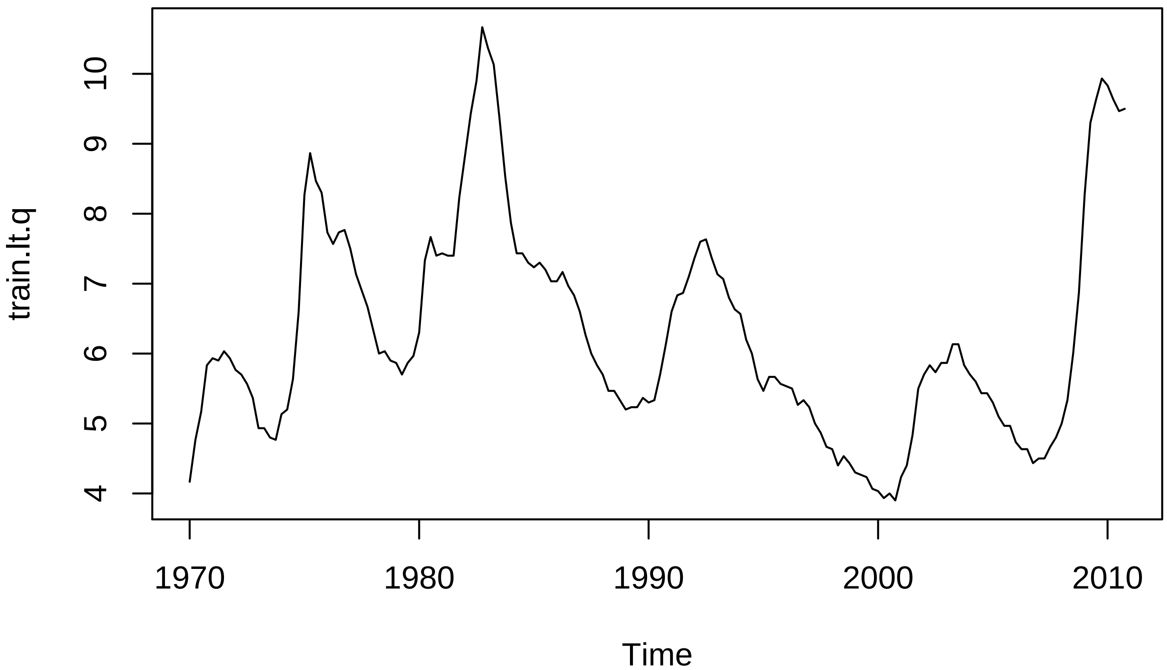


```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,1)(0,0,2)[15] with non-zero mean
## Q* = 4.4947, df = 3, p-value = 0.2128
##
## Model df: 6.   Total lags used: 9
```

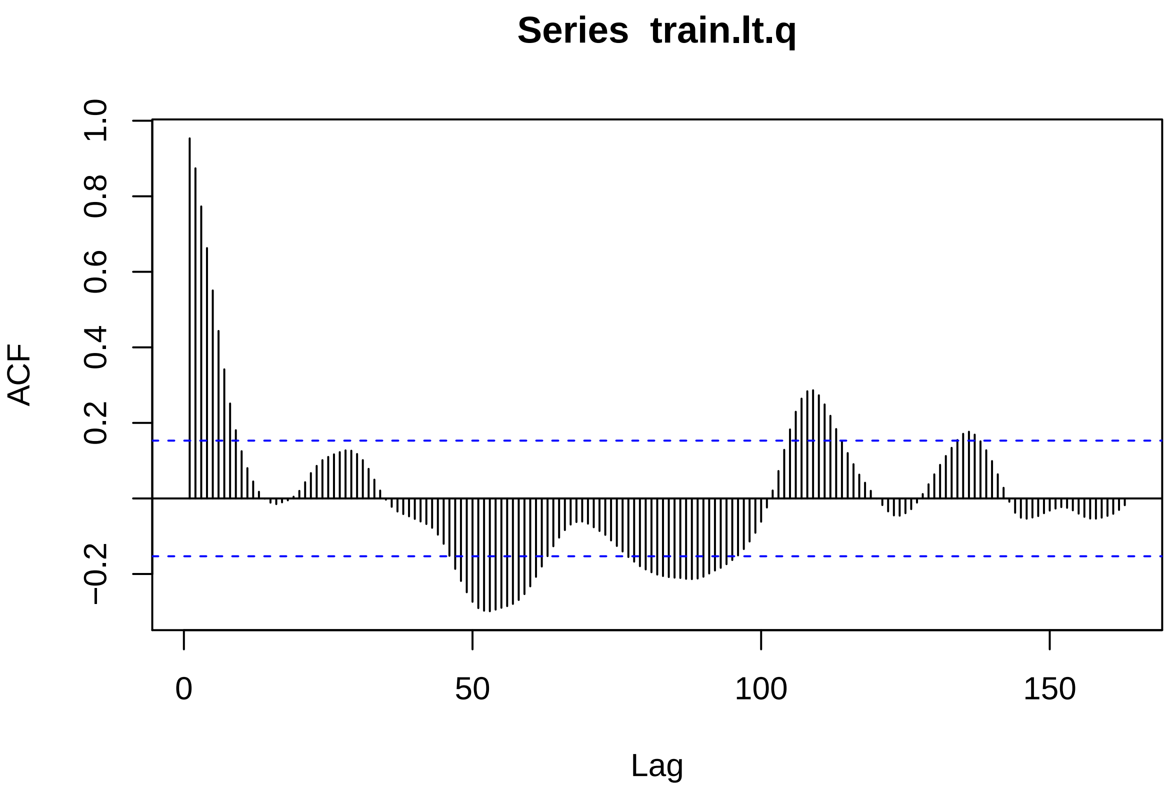
This model captures the seasonality to a fair extent. However, in the fitted values, we see that there is a lag between the actual and fitted peaks/troughs.

Try ARIMA model with quarterly aggregated data

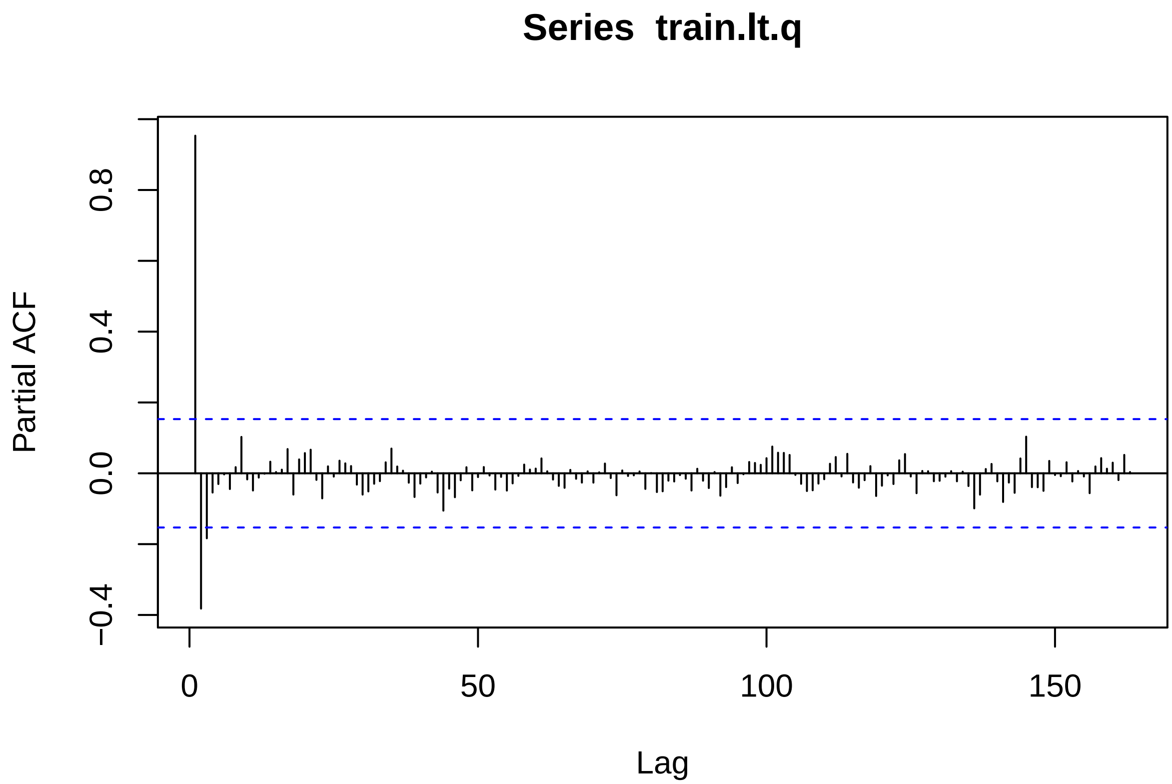
```
data.quarterly = aggregate(data.ts, nfrequency = 4, FUN = 'mean')

train.lt.q = window(data.quarterly, start = c(1970, 1), end = c(2010, 4) )
valid.lt.q = window(data.quarterly, start = c(2011, 1))

plot(train.lt.q)
```



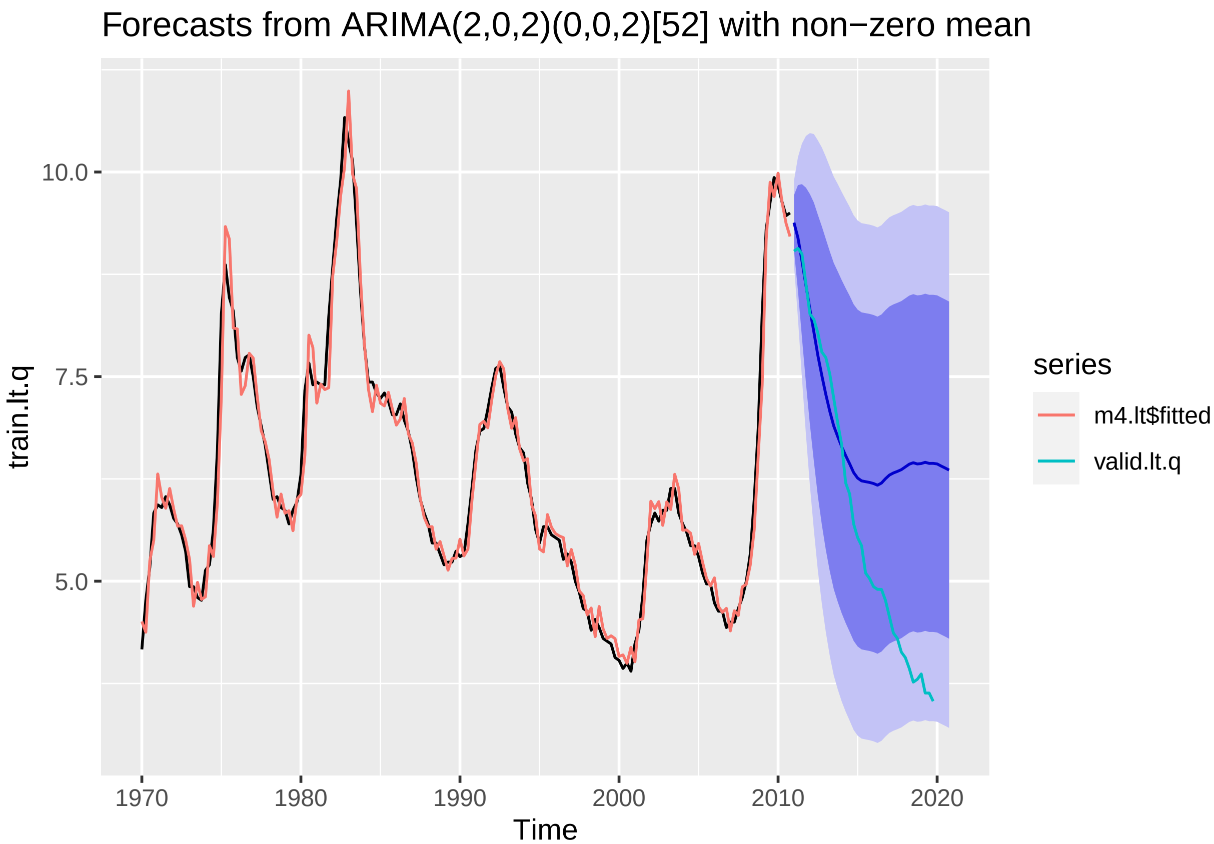

```r
Acf(train.lt.q, lag.max = 250)
```




```r
Pacf(train.lt.q, lag.max = 250)
```



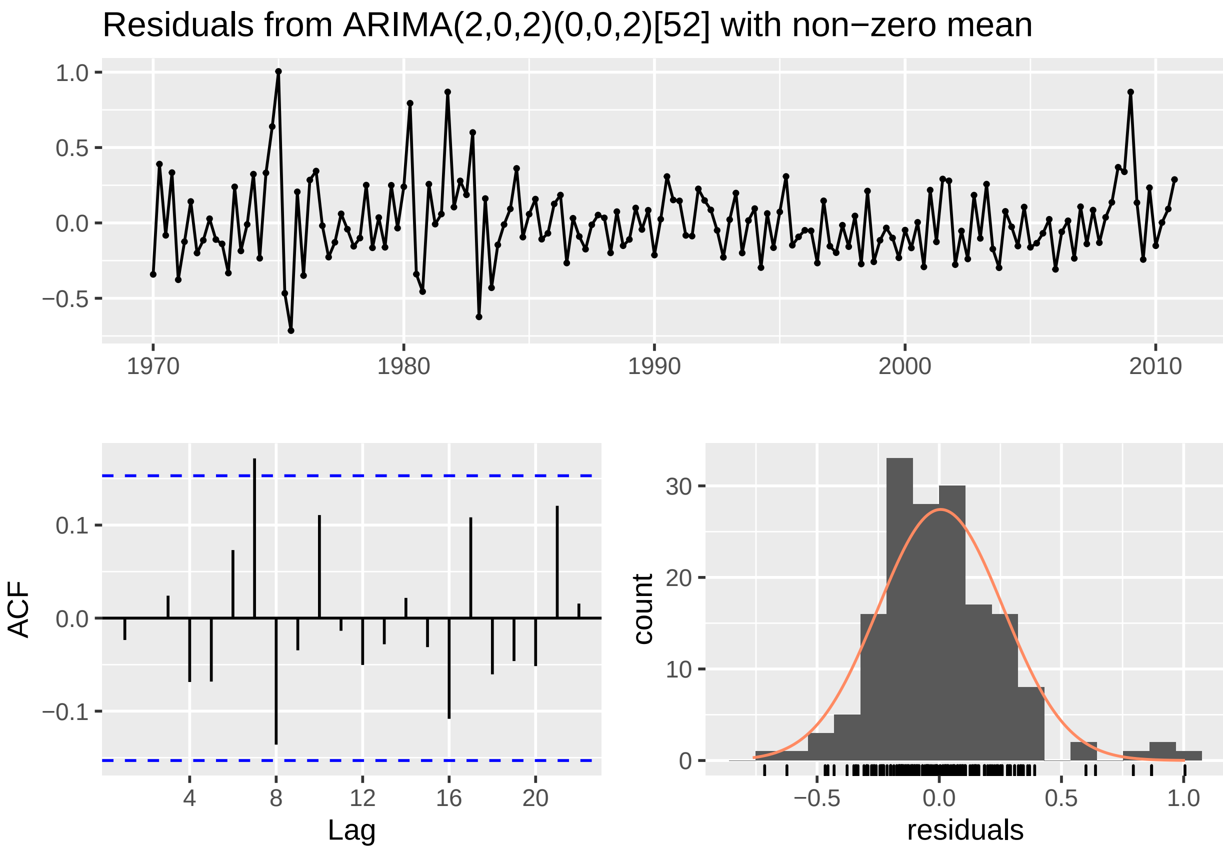

```
m4.lt = Arima( train.lt.q, order = c(2, 0, 2), seasonal = list( order = c(0, 0, 2), period = 52) )
m4.lt.forecast = forecast( m4.lt, h = 40 )

autoplot(m4.lt.forecast)+autolayer(valid.lt.q)+autolayer(m4.lt$fitted)
```




```r
accuracy(m4.lt.forecast, valid.lt.q)
```

```
##                        ME      RMSE       MAE          MPE     MAPE     MASE
## Training set  0.006059231 0.2557602 0.1914435  -0.09375283  3.04312 0.234840
## Test set     -1.053872561 1.5369290 1.1790535 -25.83530743 27.47065 1.446322
##                     ACF1 Theil's U
## Training set -0.02350153        NA
## Test set      0.94618767  11.46505
```

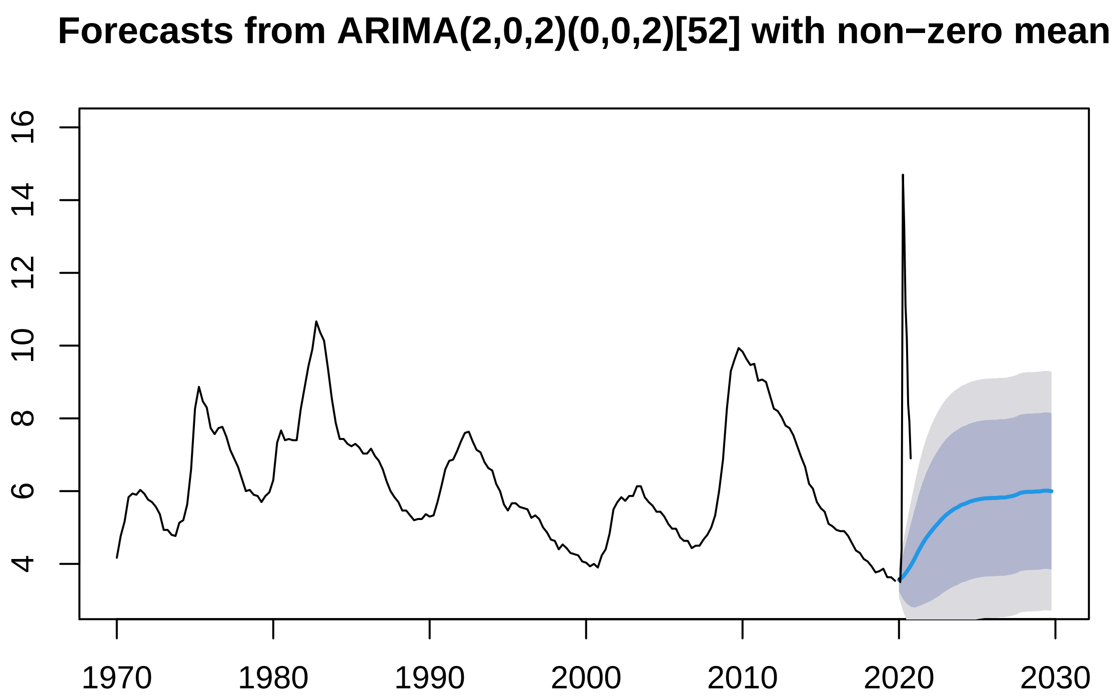
```r
checkresiduals(m4.lt.forecast)
```




```
##
## 	Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,2)(0,0,2)[52] with non-zero mean
## Q* = 13.428, df = 3, p-value = 0.003797
##
## Model df: 7.   Total lags used: 10
```

Among the two ARIMA models, we can see that the model built using the quarterly data predicts the peaks and troughs more accurately. The RMSE and MAPE for this model is slightly higher, possibly due to the presence of more data (quarterly vs. yearly). Since our goal is to predict huge increases in unemployment accurately, we will use the ARIMA model with quarterly data to predict unemployment for 2020. There is no residual autocorrelation among the errors, and they are mostly normally distributed.

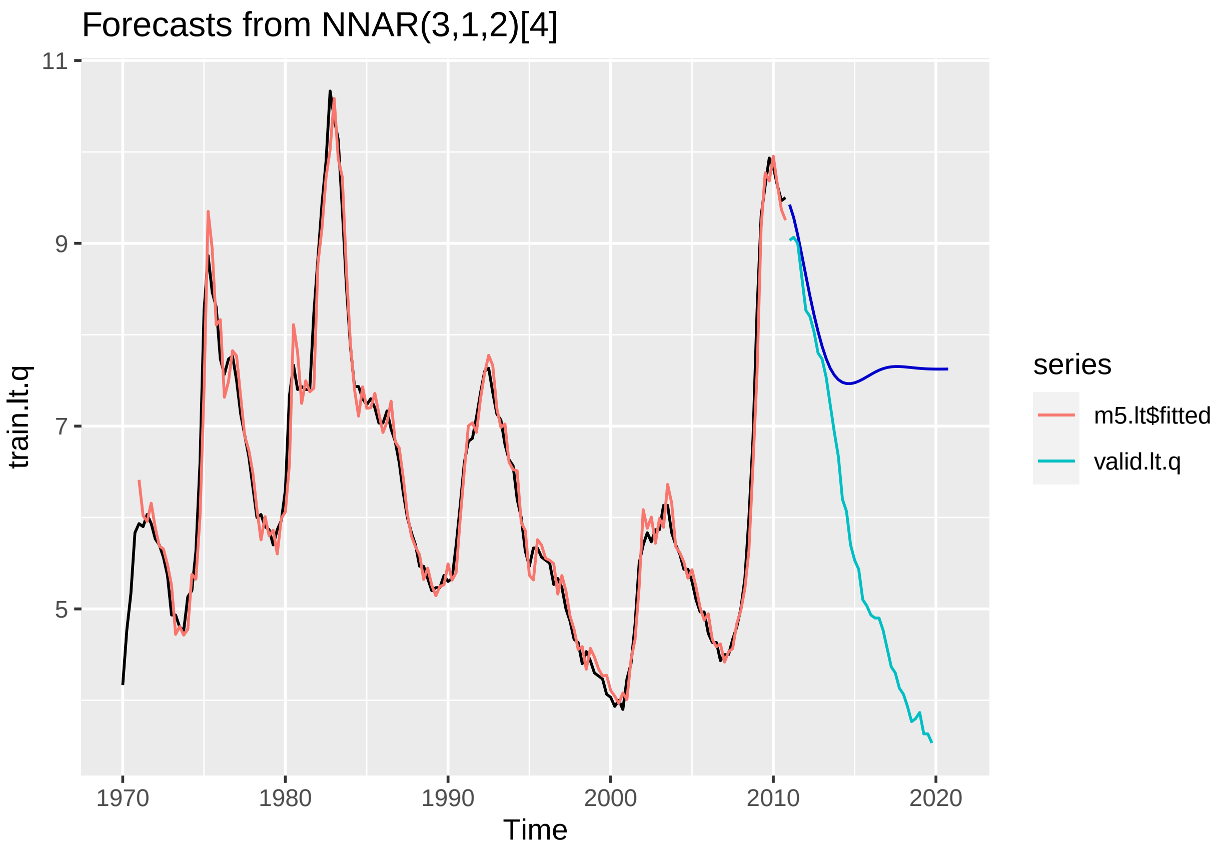
```
final.lt = Arima(window(data.quarterly, start = c(1970, 1)), order = c(2, 0, 2), seasonal = list(order =
final.lt.forecast = forecast( final.lt, h = 40)

plot(final.lt.forecast, ylim = c(3, 16))
lines(data.2020.ts)
```



Try using Neural Networks

```
# m5.lt = nnetar(train.lt.q, repeats = 30, p = 51, P = 2, MaxNWts = 1400)
# m5.lt.forecast = forecast(m5.lt, h = 40)
# autoplot(m5.lt.forecast)+autolayer(valid.lt.q)+autolayer(m5.lt$fitted)

m5.lt = nnetar(train.lt.q)
m5.lt.forecast = forecast(m5.lt, h = 40)
autoplot(m5.lt.forecast)+autolayer(valid.lt.q)+autolayer(m5.lt$fitted)
```

```
## Warning: Removed 4 row(s) containing missing values (geom_path).
```



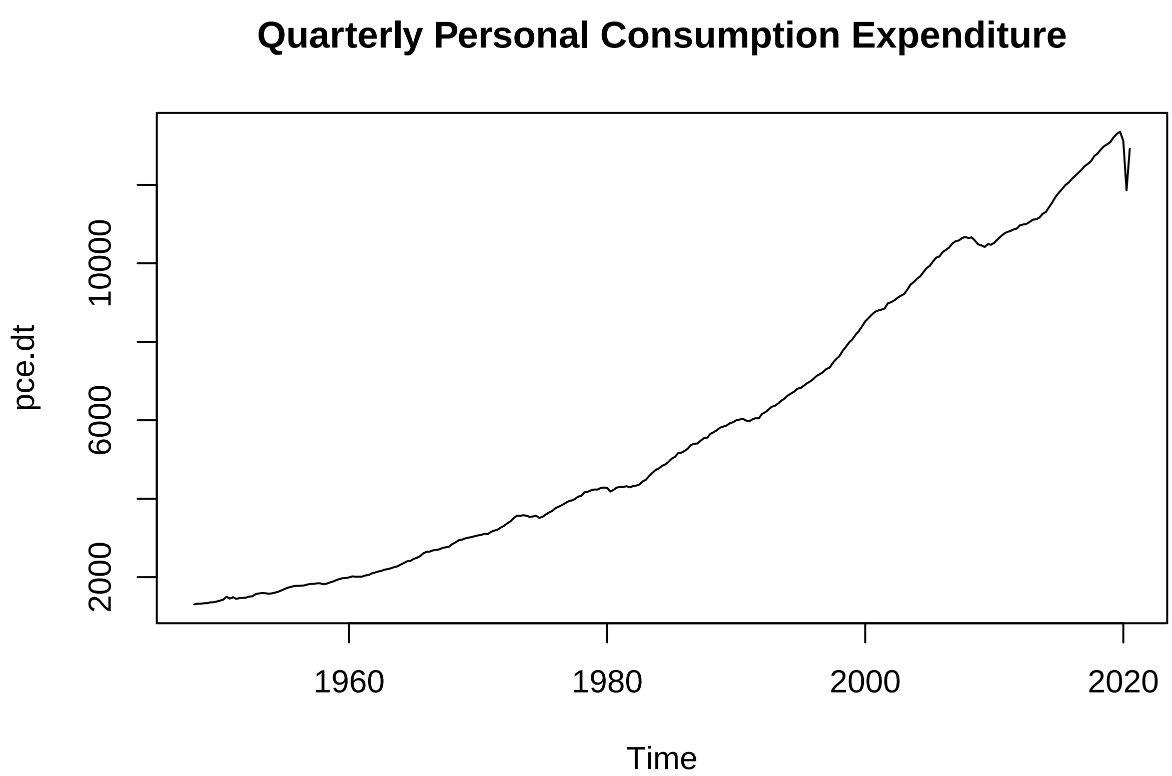
Incorporate External Variables

1. Personal Consumption Expenditure

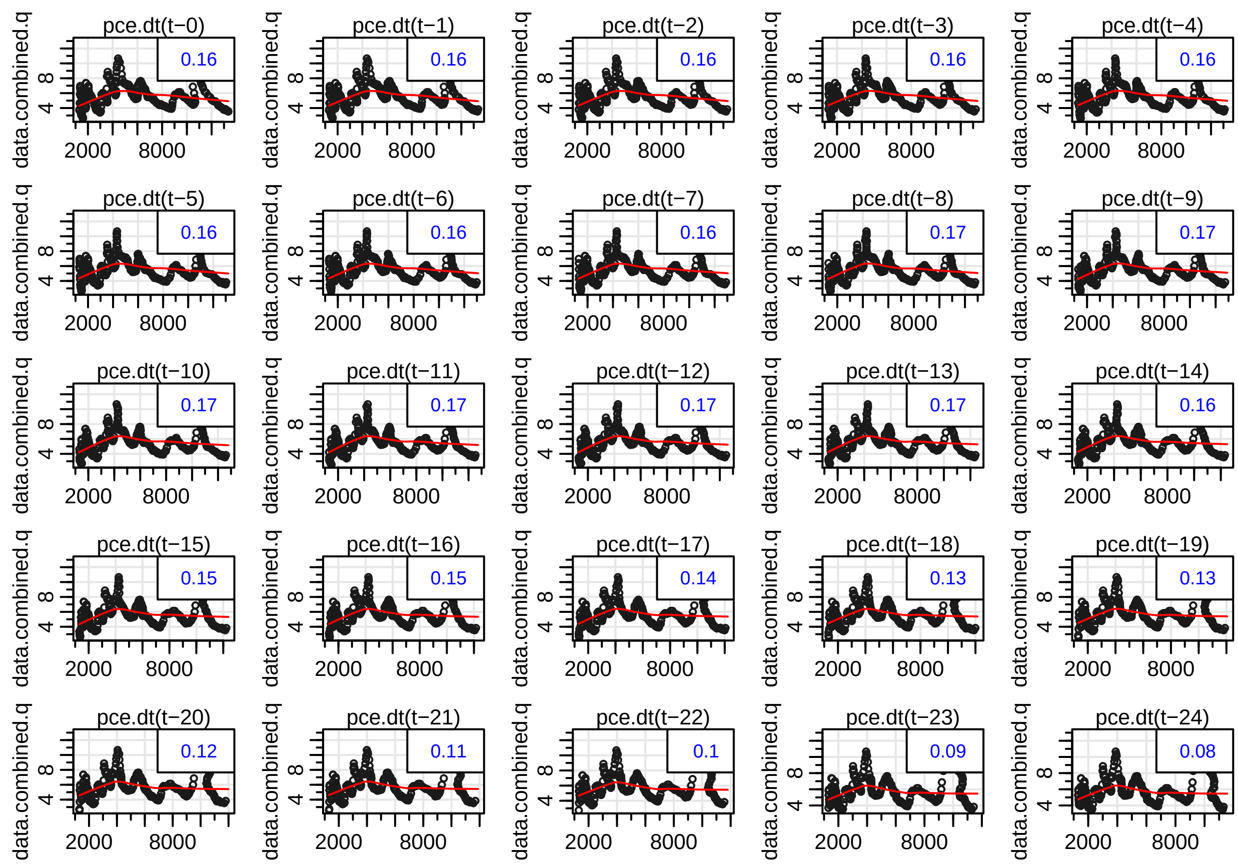
```r
library(astsa)
```

```
##
## Attaching package: 'astsa'

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

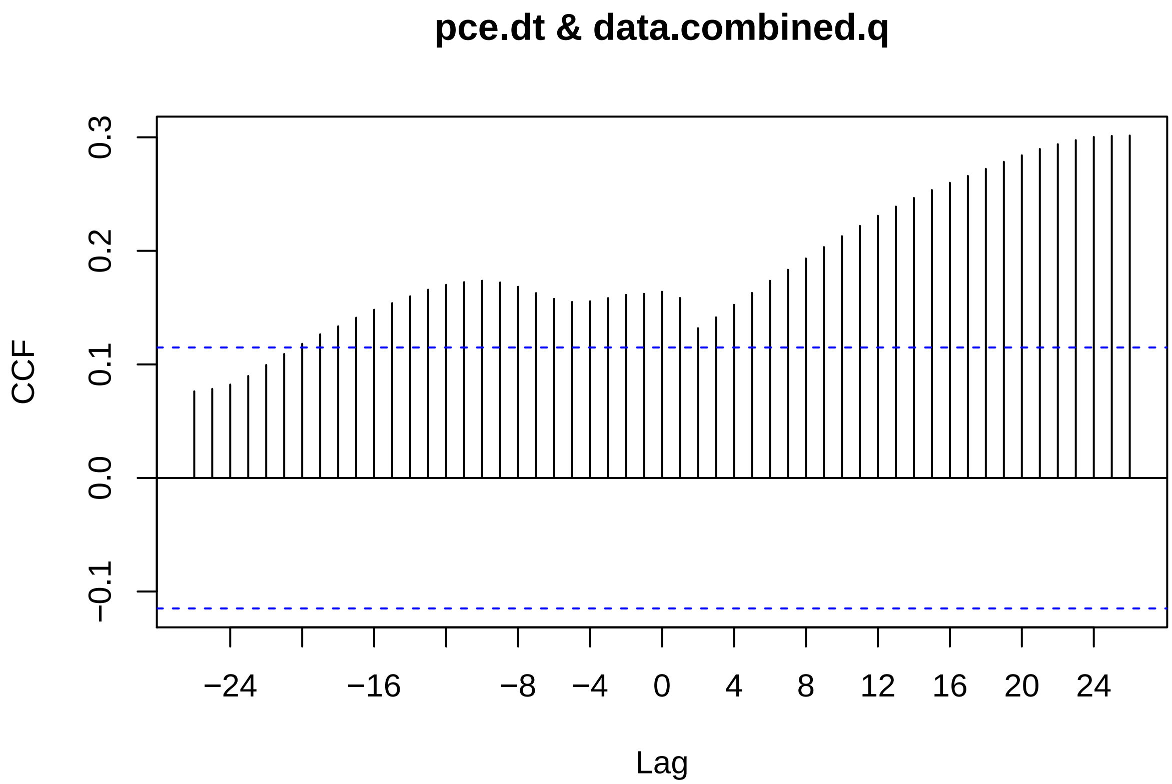
```r
data.combined.q = aggregate(data.combined, nfrequency = 4, FUN = 'mean')

pce = readxl::read_excel('PCECC96.xls')
pce.ts = ts(pce$PCECC96, start = c(1947, 1), frequency = 4 )
pce.dt = window(pce.ts, start = c(1948, 1))
plot(pce.dt, main = 'Quarterly Personal Consumption Expenditure')
```



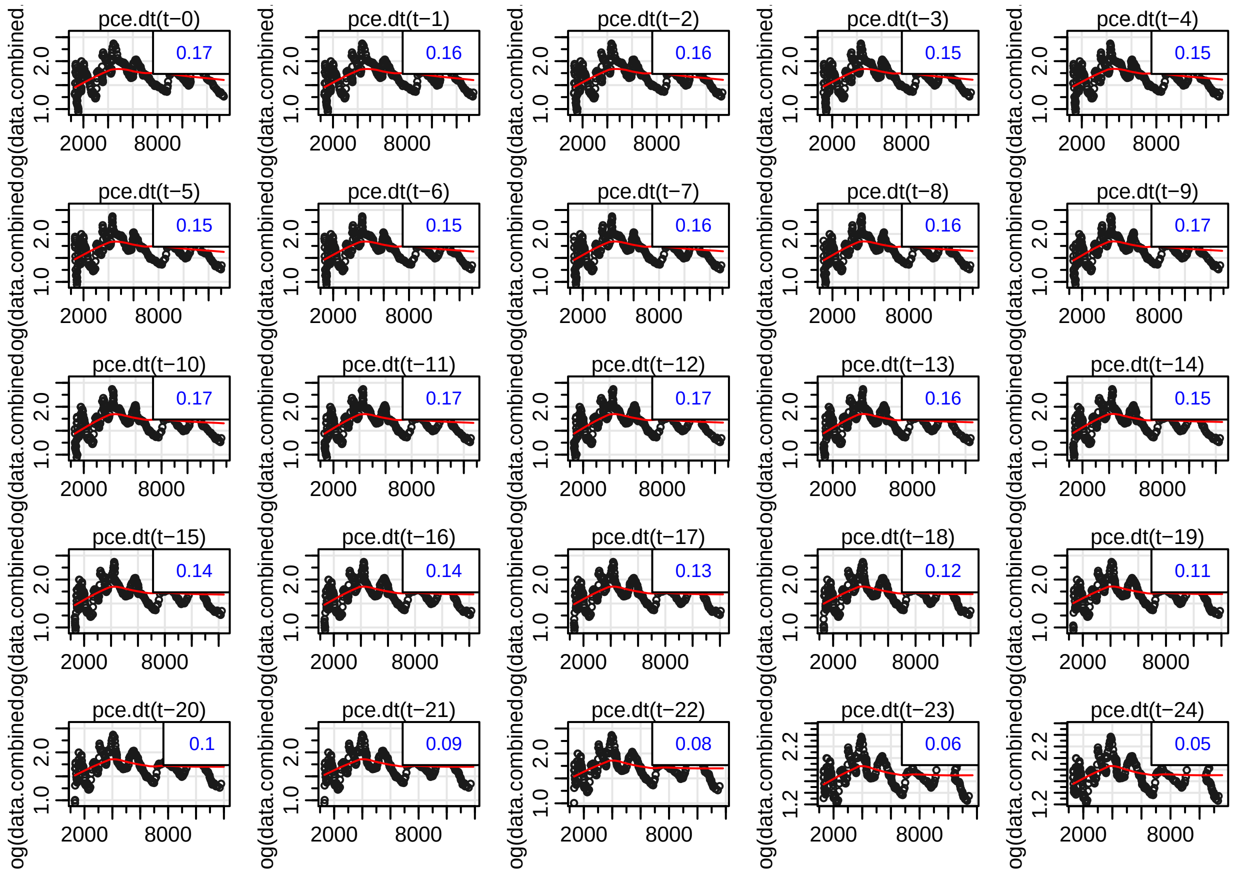

```r
lag2.plot(pce.dt, data.combined.q, 24)
```



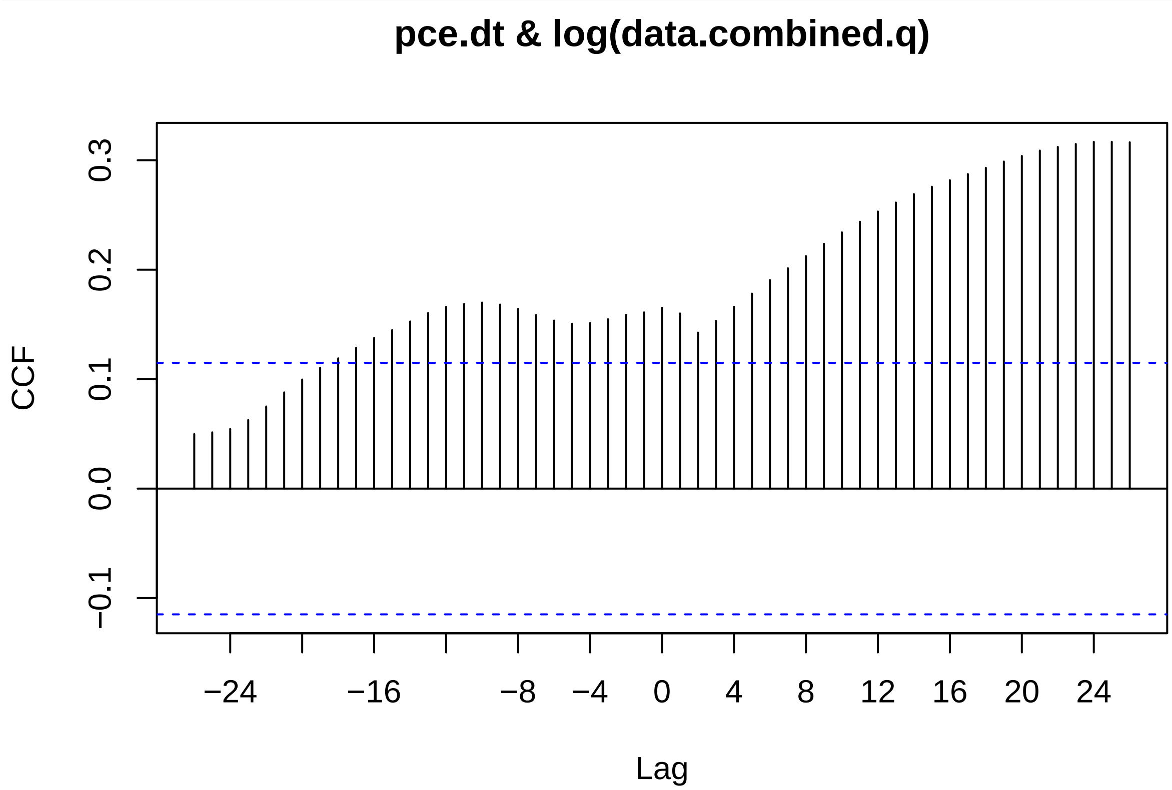
```r
Ccf(pce.dt, data.combined.q, 26)
```




```r
#The highest correlation is at lag-10

lag2.plot(pce.dt, log(data.combined.q), 24)
```




```
Ccf(pce.dt, log(data.combined.q), 26)
```



```r
newdata = ts.intersect( UR = data.combined.q, Pce = pce.dt, lagPce10 = lag(pce.dt, -10), lagUR52 = lag(
head(newdata)
```

```
##                UR      Pce lagPce10  lagUR52
## 1961 Q3 6.766667 2053.774 1923.675 3.733333
## 1961 Q4 6.200000 2095.084 1953.384 3.666667
## 1962 Q1 5.633333 2117.277 1973.791 3.766667
## 1962 Q2 5.533333 2143.306 1976.014 3.833333
## 1962 Q3 5.566667 2160.580 1994.918 4.666667
## 1962 Q4 5.533333 2191.150 2020.082 5.866667
```

```r
m1.ex = tslm(UR ~ Pce + lagPce10 + lagUR52, data = window(newdata, start = c(1970,1)))
summary(m1.ex)
```

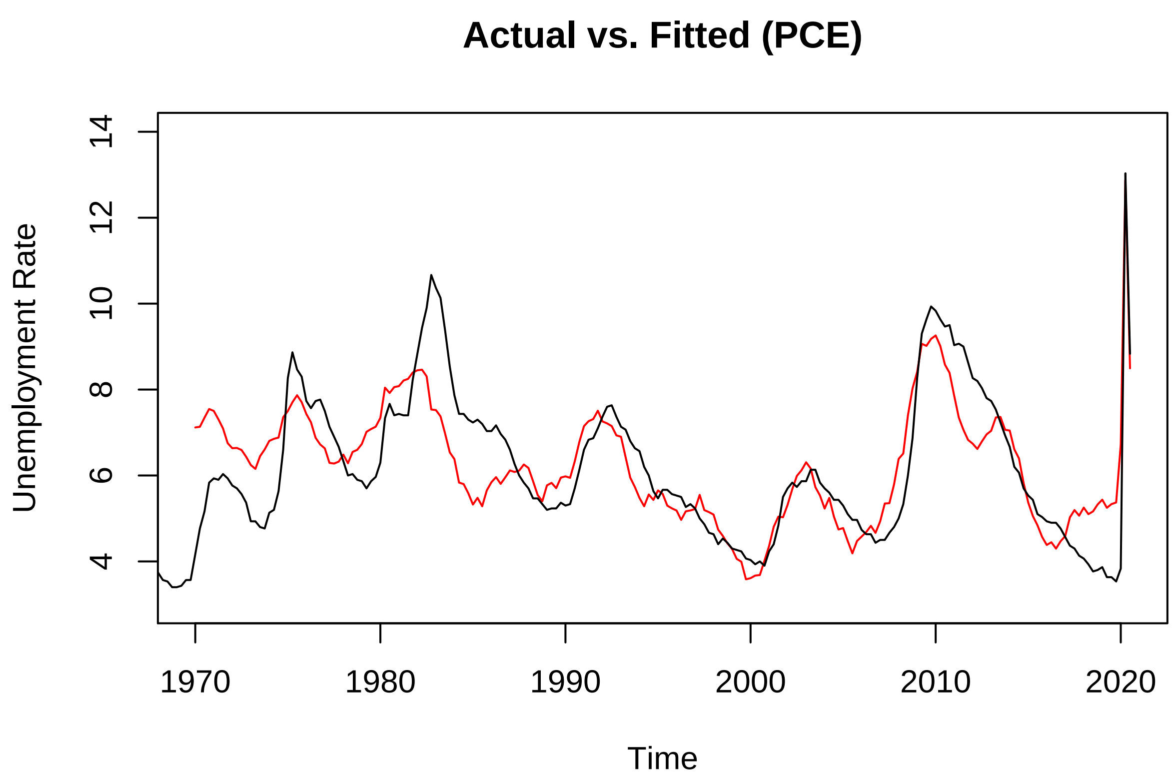
```
##
## Call:
## tslm(formula = UR ~ Pce + lagPce10 + lagUR52, data = window(newdata,
##     start = c(1970, 1)))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.95161 -0.64667  0.01287  0.58399  3.13125
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.029675   0.353226  25.563  < 2e-16 ***
## Pce         -0.004395   0.000292 -15.051  < 2e-16 ***
## lagPce10     0.004458   0.000303  14.712  < 2e-16 ***
## lagUR52     -0.181267   0.054687  -3.315  0.00109 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.029 on 199 degrees of freedom
## Multiple R-squared:  0.6263, Adjusted R-squared:  0.6207
## F-statistic: 111.2 on 3 and 199 DF,  p-value: < 2.2e-16
```

```r
accuracy( m1.ex$fitted.values, window(data.combined.q, start = c(1970,1)) )
```

```
##                     ME    RMSE       MAE       MPE     MAPE      ACF1 Theil's U
## Test set -1.522343e-17 1.01856 0.7948877 -2.595435 13.32527 0.9056362 0.9793355
```

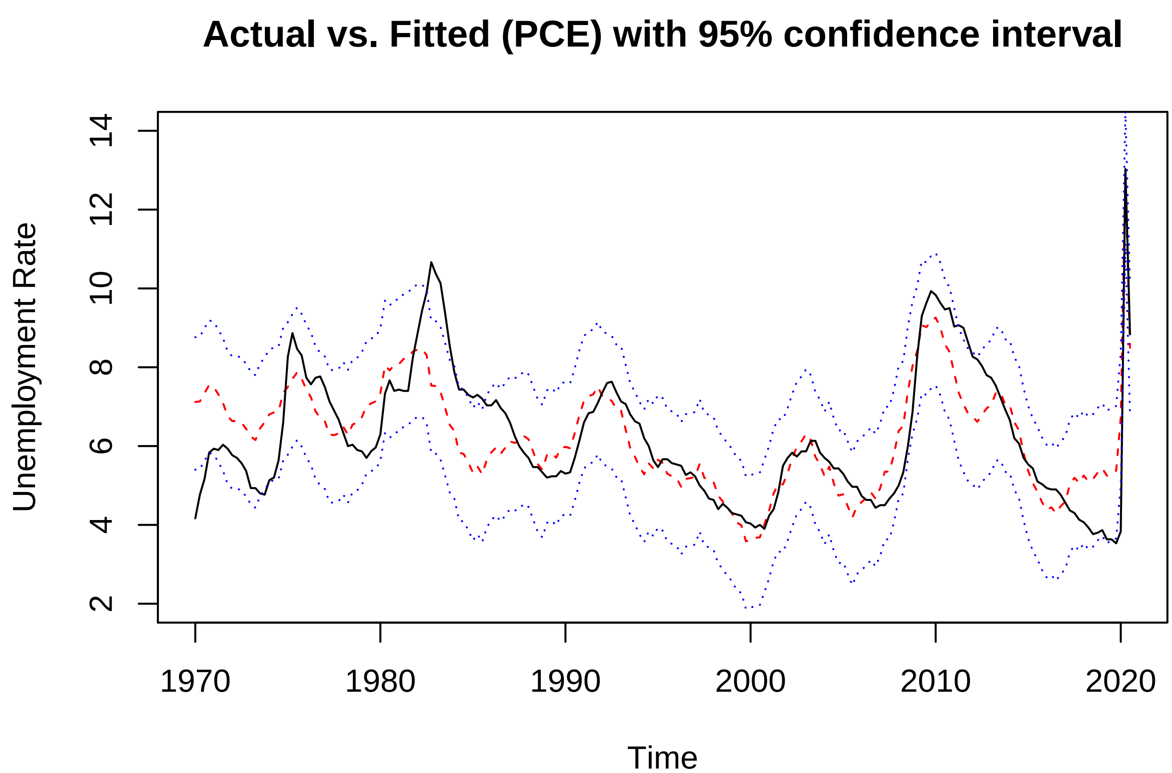
```r
e = m1.ex$fitted.values - window(data.combined.q, start = c(1970,1))
m = quantile(e, prob = c(0.05, 0.95))

plot(m1.ex$fitted.values, ylim = c(3, 14), col = 'red', ylab = "Unemployment Rate", main =
"Actual vs. Fitted (PCE)")+lines(window(data.combined.q), start = c(1970,1))
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "start" is not a
## graphical parameter
```




```
## integer(0)
```

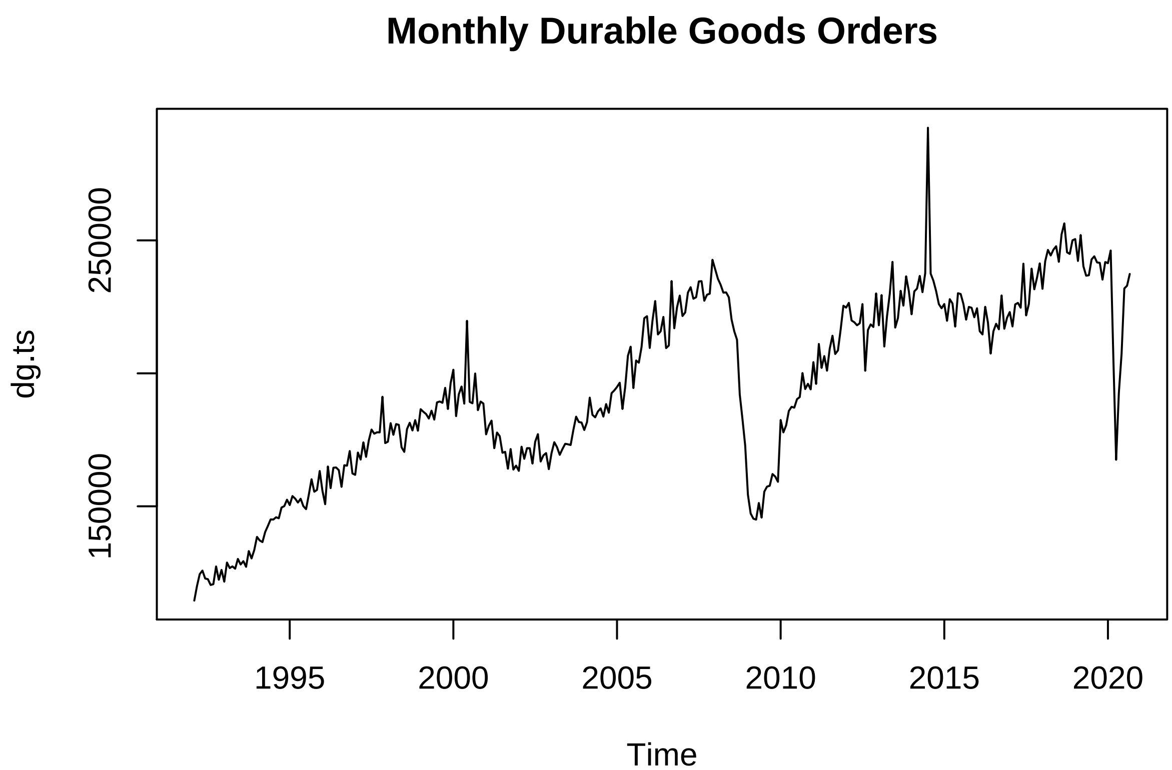
```
plot(m1.ex$fitted.values, ylim = c(2, 14), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. F
lines( m1.ex$fitted.values + m[1], col = 'blue', lty = 3 ) +
lines( m1.ex$fitted.values + m[2], col = 'blue', lty = 3 )
```



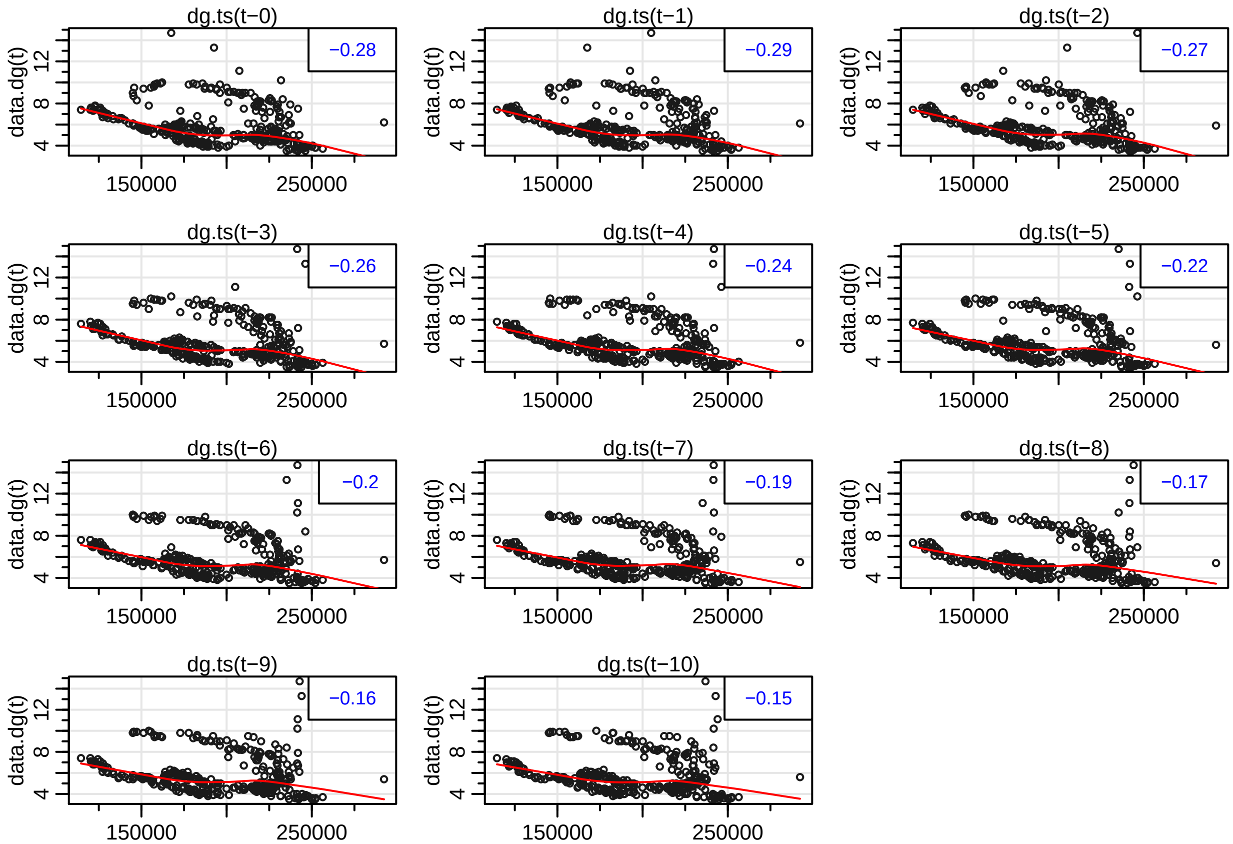
```
## integer(0)
```

2. Durable Goods Orders

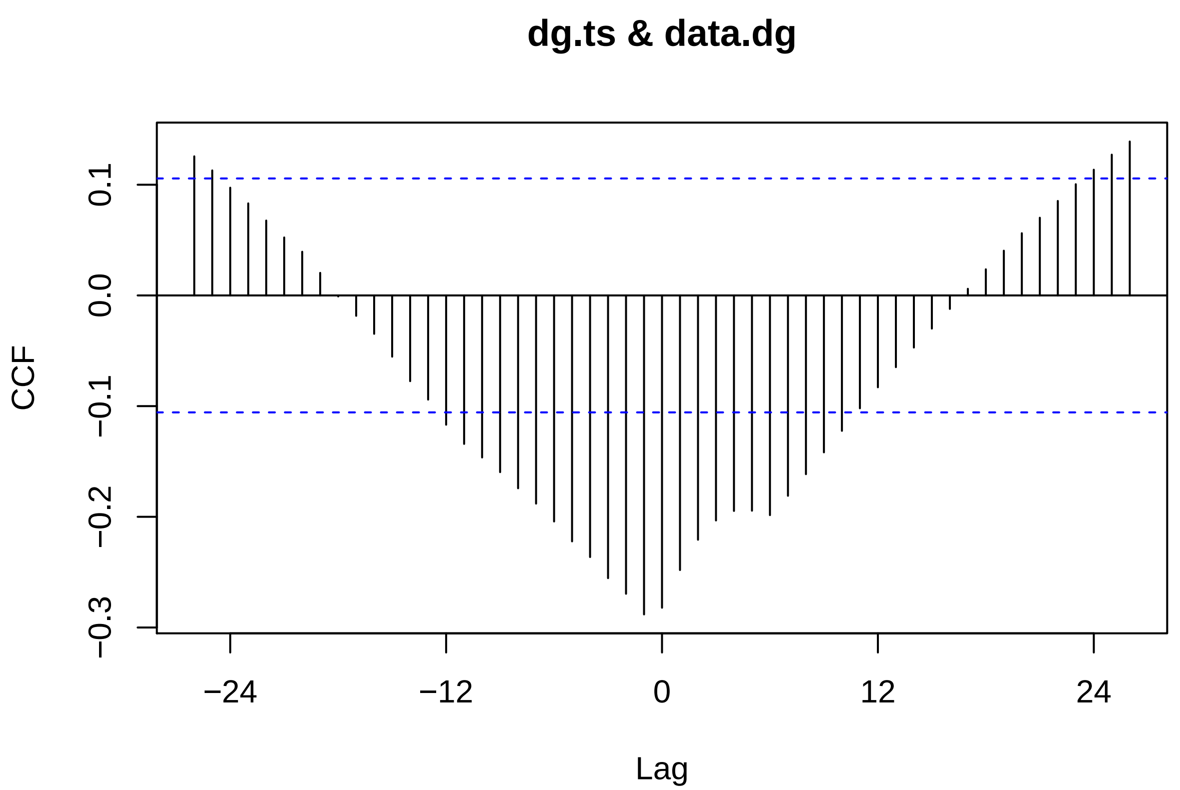
```r
dg = readxl::read_excel('DGORDER.xls')

dg.ts = ts(dg$DGORDER, start = c(1992, 2), frequency = 12)
plot(dg.ts, main = 'Monthly Durable Goods Orders')
```



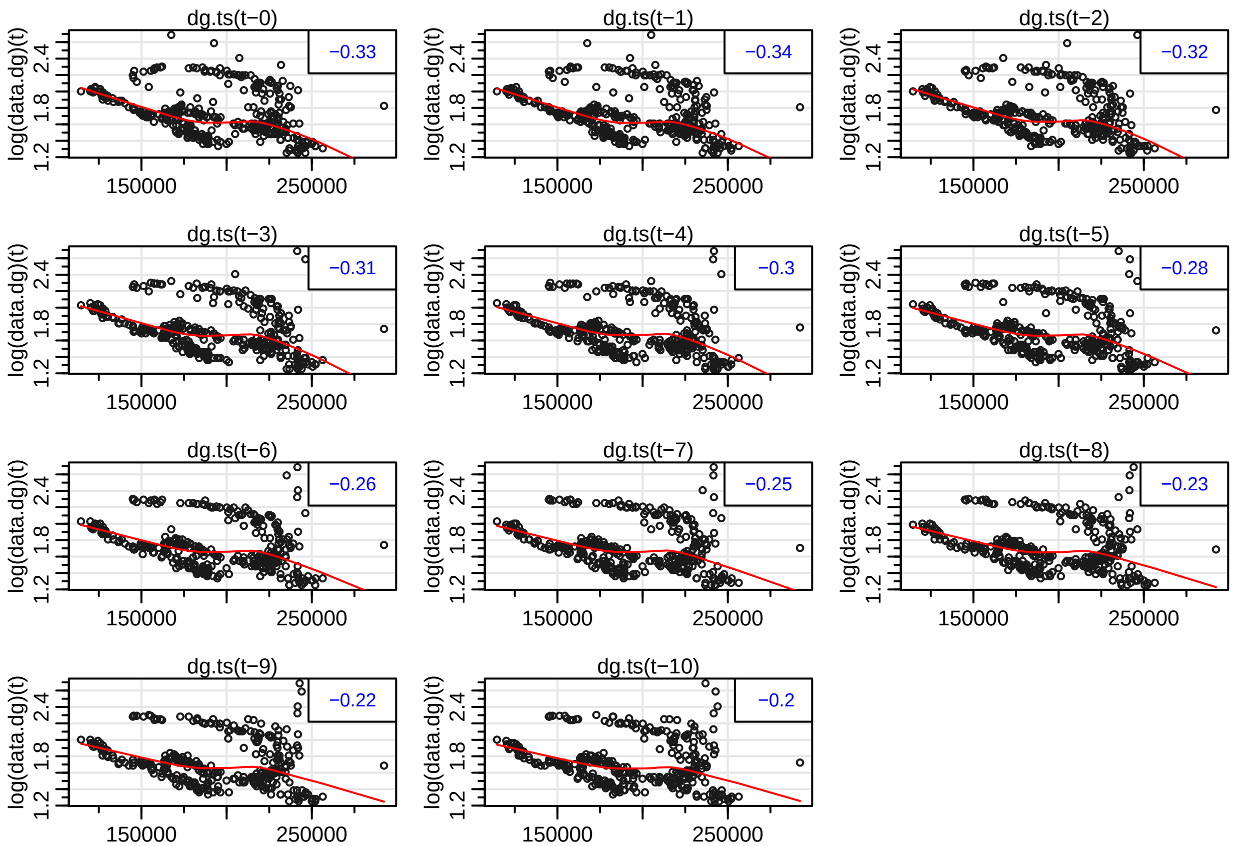

```r
data.dg = window(data.combined, start = c(1992, 2))

lag2.plot(dg.ts, data.dg, 10)
```



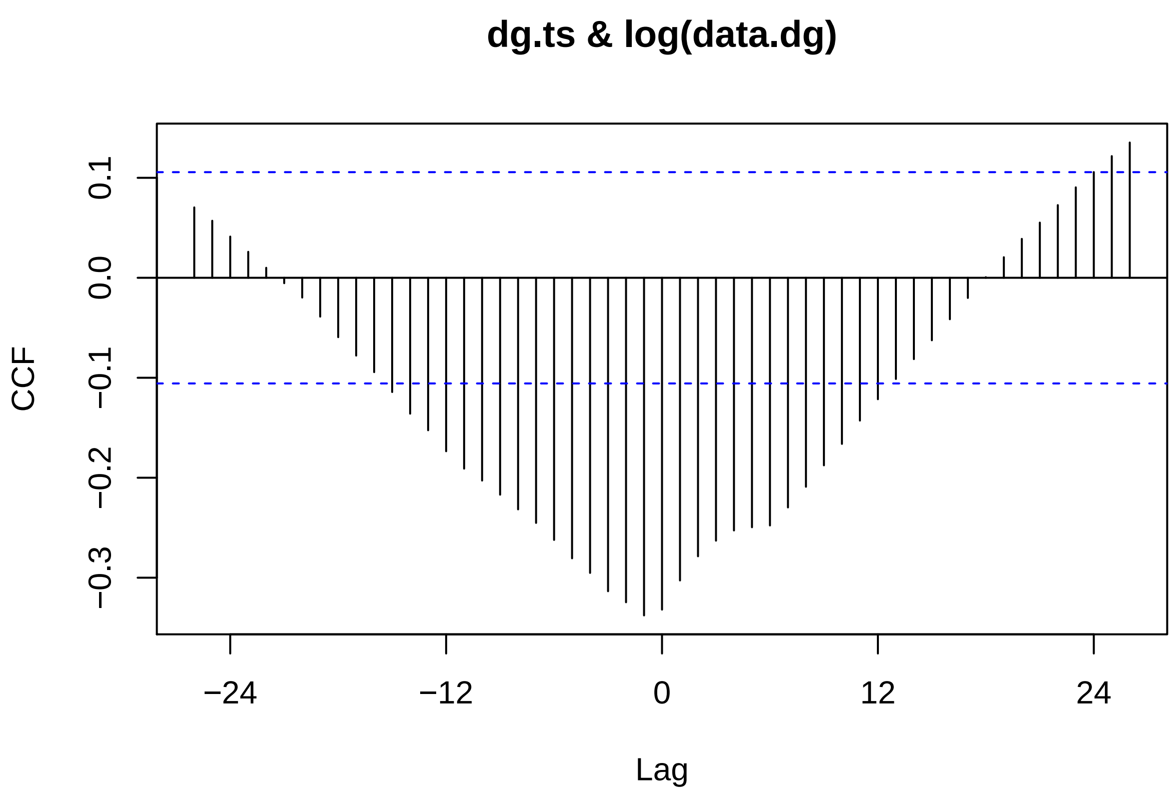

```
Ccf(dg.ts, data.dg, 26)
```



```r
lag2.plot(dg.ts, log(data.dg), 10)
```




```r
Ccf(dg.ts, log(data.dg), 26)
```




```r
#There is a better correlation between lagged DGO and log unemployment rate

newdata_2 = ts.intersect( UR = log(data.dg), DG = dg.ts, lagDG1 = lag(dg.ts, -1) )
head(newdata_2)
```

```
##                UR     DG lagDG1
## Mar 1992 2.001480 120025 114535
## Apr 1992 2.001480 124470 120025
## May 1992 2.028148 125822 124470
## Jun 1992 2.054124 122834 125822
## Jul 1992 2.041220 122590 122834
## Aug 1992 2.028148 120411 122590
```

```r
m2.ex = tslm(UR ~ DG + lagDG1 , data = newdata_2 )
summary(m2.ex)
```

```
##
## Call:
## tslm(formula = UR ~ DG + lagDG1, data = newdata_2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39972 -0.18136 -0.06779  0.10560  0.98116
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.260e+00  8.178e-02  27.629   <2e-16 ***
## DG          -1.769e-07  1.558e-06  -0.114     0.91
## lagDG1      -2.553e-06  1.549e-06  -1.647     0.10
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2633 on 340 degrees of freedom
## Multiple R-squared:  0.1149, Adjusted R-squared:  0.1097
## F-statistic: 22.07 on 2 and 340 DF,  p-value: 9.695e-10
```

```
accuracy( m2.ex$fitted.values, window(log(data.dg), start = c(1992,1)) )
```

```
## Warning in window.default(x, ...): 'start' value not changed
```

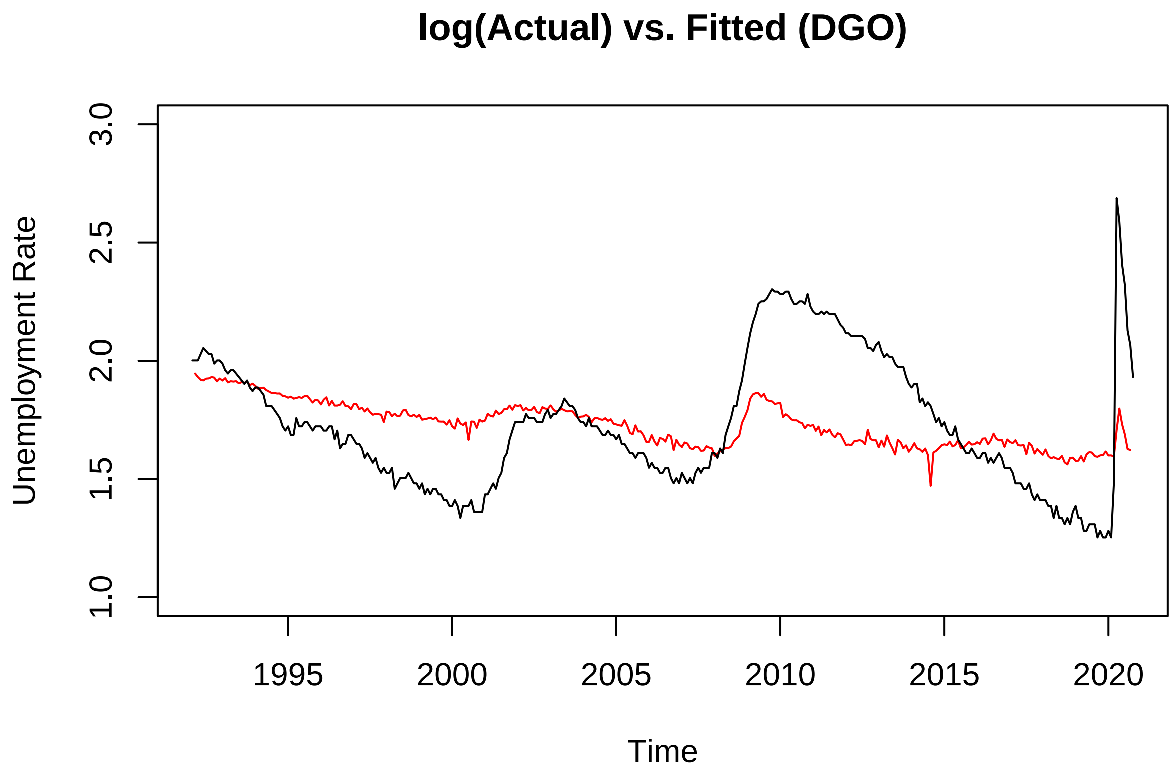
```
##                    ME      RMSE       MAE       MPE     MAPE     ACF1 Theil's U
## Test set 4.77467e-17 0.2621315 0.2064206 -2.173163 11.84757 0.959998  3.047478
```

```
e = m2.ex$fitted.values - window(log(data.dg), start = c(1992,1))
```

```
## Warning in window.default(x, ...): 'start' value not changed
```

```
m = quantile(e, prob = c(0.05, 0.95))

plot(m2.ex$fitted.values, col = 'red', ylim = c(1,3), ylab = "Unemployment Rate", main = "log(Actual) vs
```

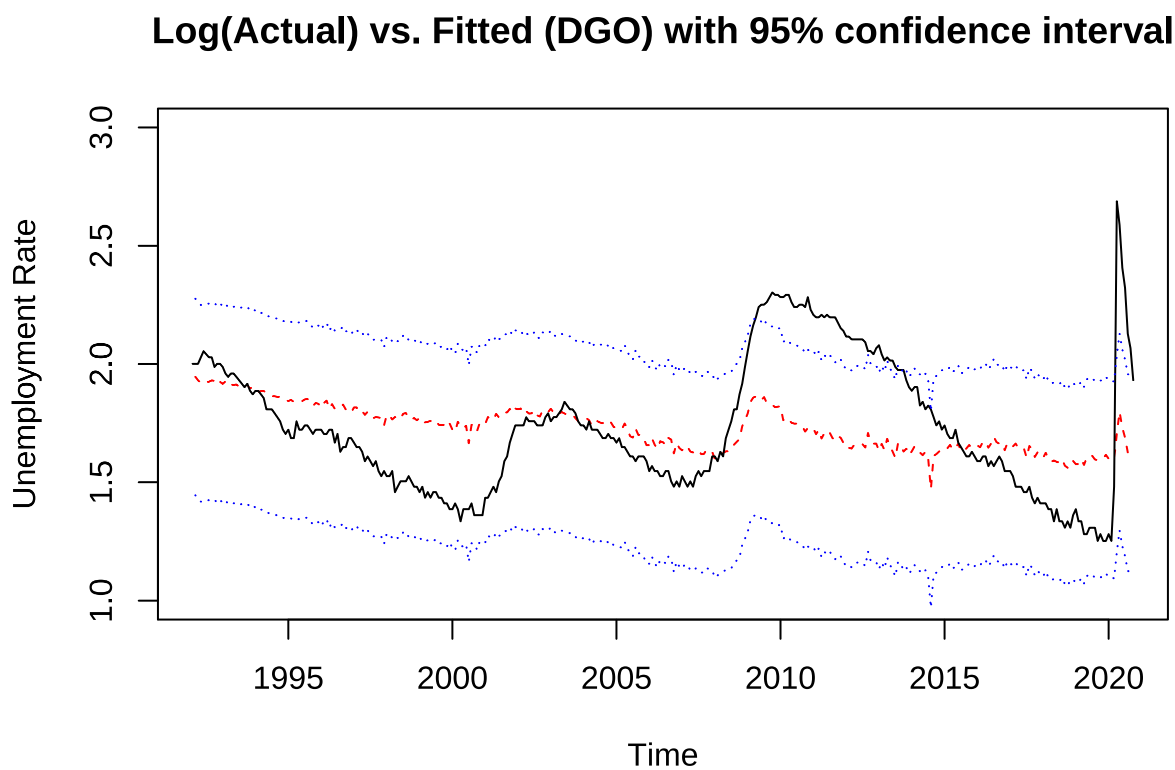
```
## Warning in window.default(x, ...): 'start' value not changed
```




```
## integer(0)
```

```
plot(m2.ex$fitted.values, ylim = c(1, 3), col = 'red', ylab = "Unemployment Rate", main = "Log(Actual) v
lines( m2.ex$fitted.values + m[1], col = 'blue', lty = 3 ) +
lines( m2.ex$fitted.values + m[2], col = 'blue', lty = 3 )
```

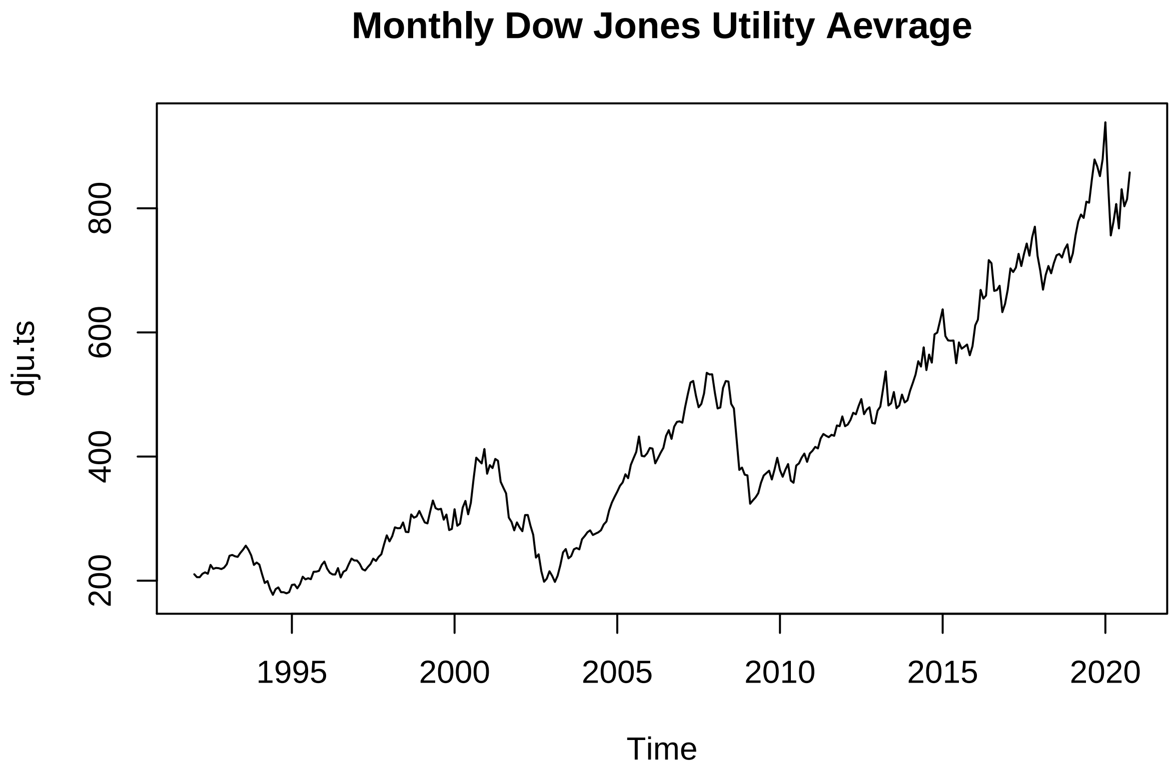
```
## Warning in window.default(x, ...): 'start' value not changed
```



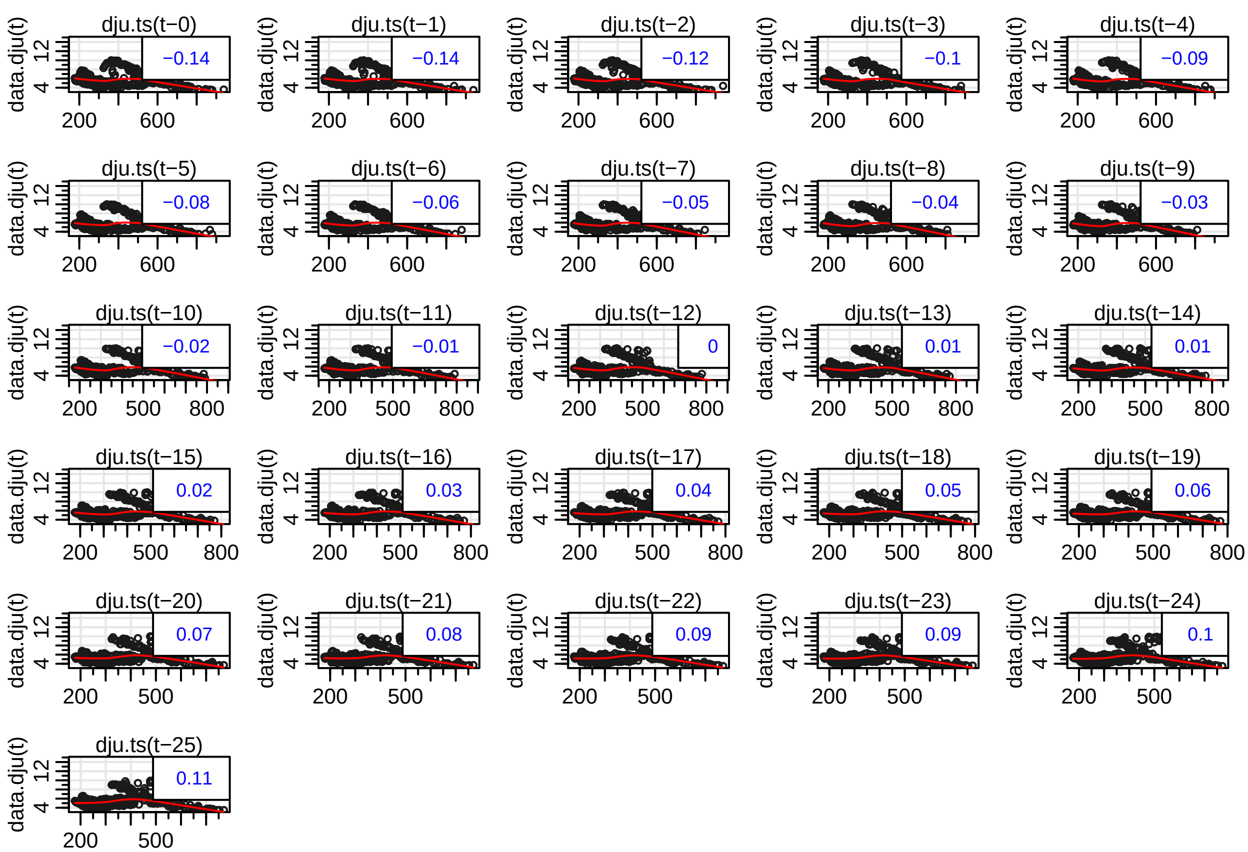

```
## integer(0)
```

3. Dow-Jones Industrial Average

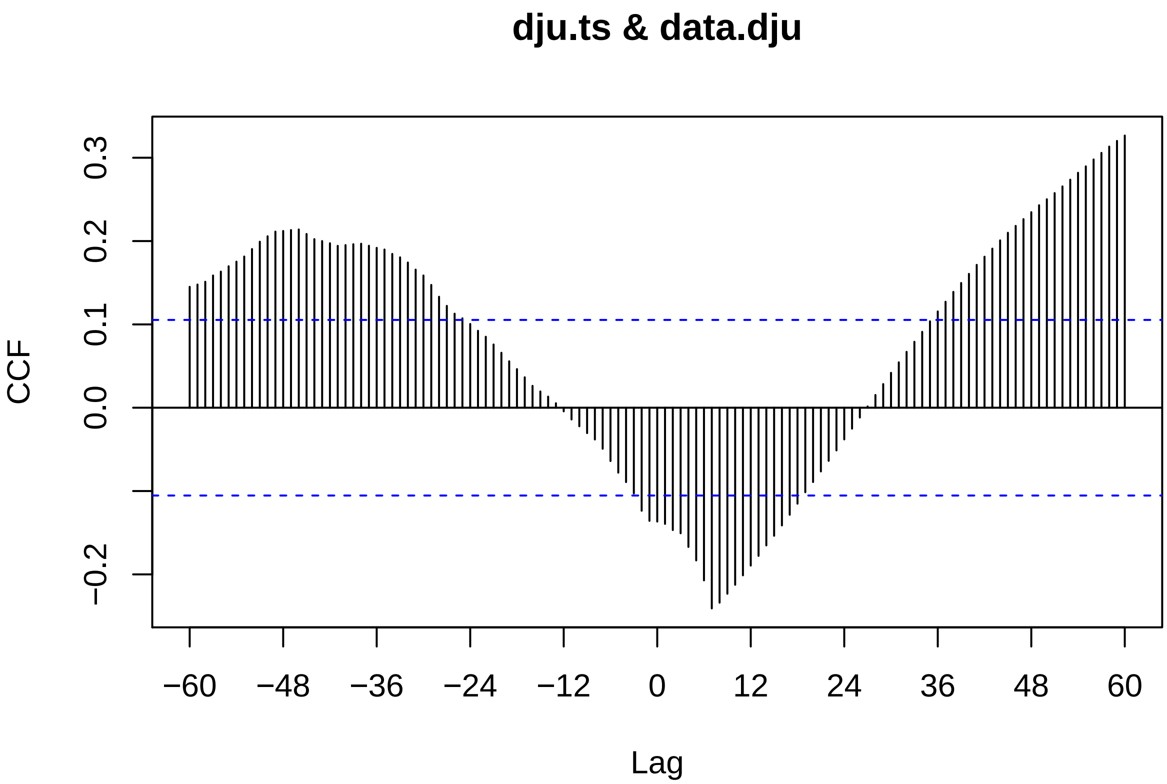
```
dju = read.csv('^DJU.csv')
dju.ts = ts(dju$Adj.Close, start = c(1992, 1), end = c(2020, 10), frequency = 12)
plot(dju.ts, main = 'Monthly Dow Jones Utility Aevrage')
```



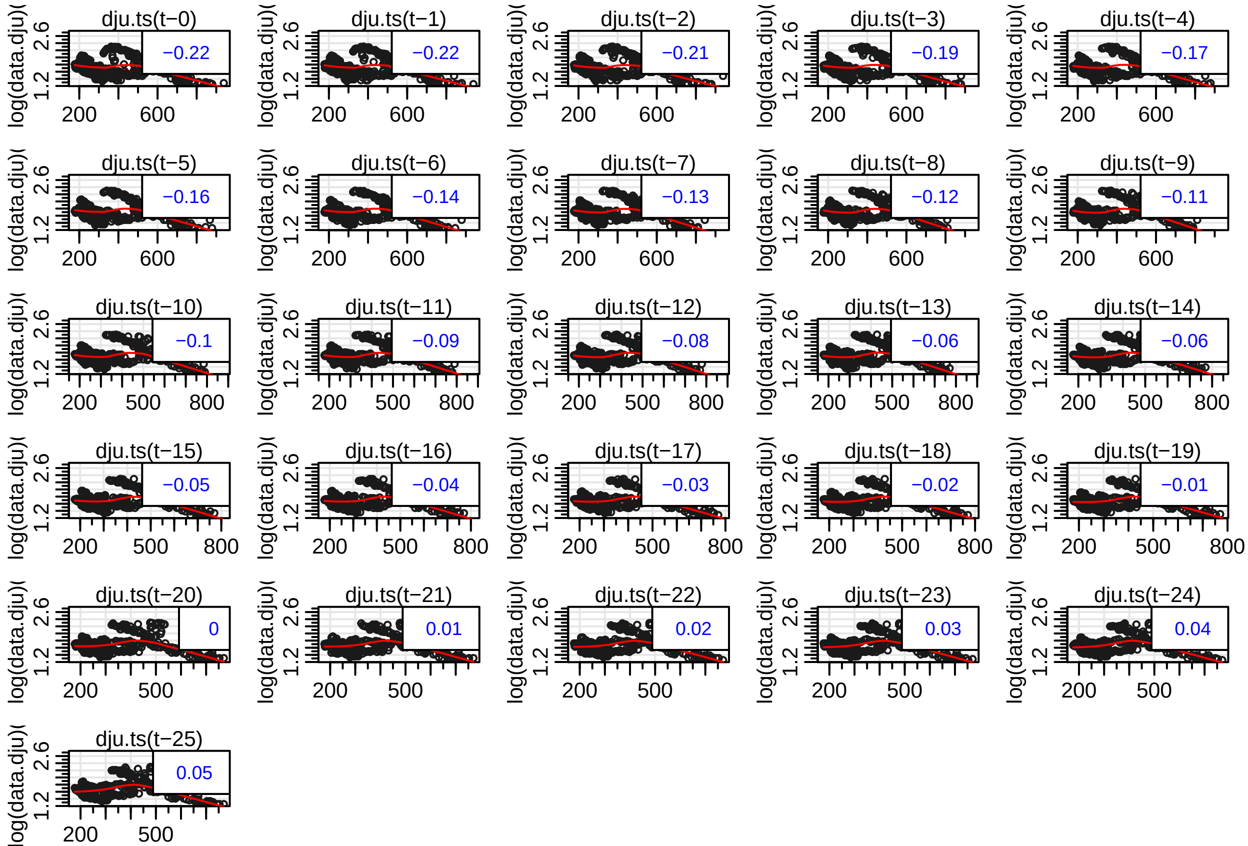

```
data.dju = window(data.combined, start = c(1992, 1), end = c(2020, 10))

lag2.plot(dju.ts, data.dju, 25)
```



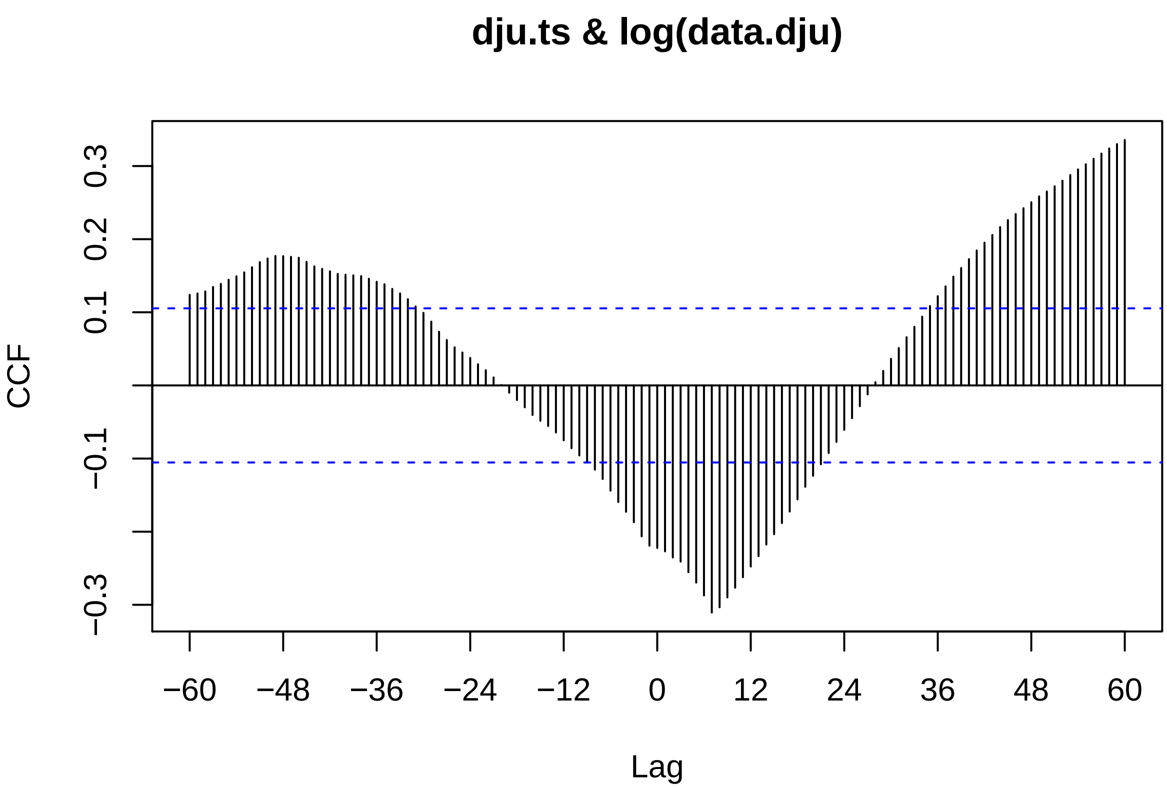
```r
Ccf(dju.ts, data.dju, 60)
```




```r
lag2.plot(dju.ts, log(data.dju), 25)
```




```
Ccf(dju.ts, log(data.dju), 60)
```



```r
newdata_3 = ts.intersect( UR = data.dju, DJU = dju.ts, lagDJU48 = lag(dju.ts, -48), lagUR216 = lag(data
head(newdata_3)
```

```
##           UR    DJU lagDJU48 lagUR216
## Jan 1996 5.6 230.85   210.38      6.4
## Feb 1996 5.5 219.40   205.62      6.3
## Mar 1996 5.5 212.76   205.62      6.3
## Apr 1996 5.6 210.10   211.07      6.1
## May 1996 5.6 209.96   213.45      6.0
## Jun 1996 5.3 220.30   211.13      5.9
```

```r
m3.ex = tslm(UR ~ DJU + lagDJU48 + lagUR216, data = newdata_3 )
summary(m3.ex)
```

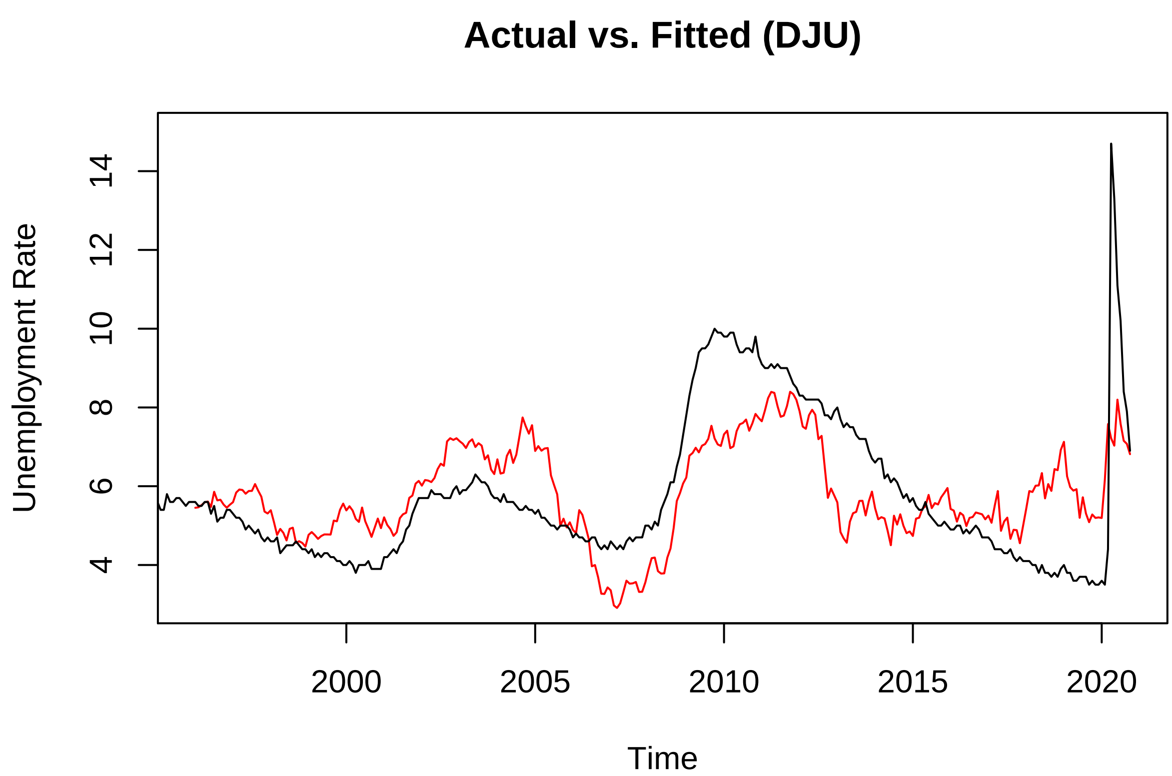
```
##
## Call:
## tslm(formula = UR ~ DJU + lagDJU48 + lagUR216, data = newdata_3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1786 -0.9307 -0.3609  1.0663  7.4840
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.3231799  0.7025402   4.730 3.49e-06 ***
## DJU         -0.0084890  0.0008739  -9.713  < 2e-16 ***
## lagDJU48     0.0146139  0.0010709  13.647  < 2e-16 ***
## lagUR216     0.1586048  0.0754207   2.103   0.0363 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.47 on 294 degrees of freedom
## Multiple R-squared:  0.3974, Adjusted R-squared:  0.3912
## F-statistic: 64.62 on 3 and 294 DF,  p-value: < 2.2e-16
```

```r
accuracy( m3.ex$fitted.values, window(data.dju, start = c(1996,1)) )
```

```
##                    ME     RMSE      MAE       MPE     MAPE     ACF1 Theil's U
## Test set 1.202666e-17 1.460481 1.159927 -5.326943 20.77183 0.876564  1.967202
```

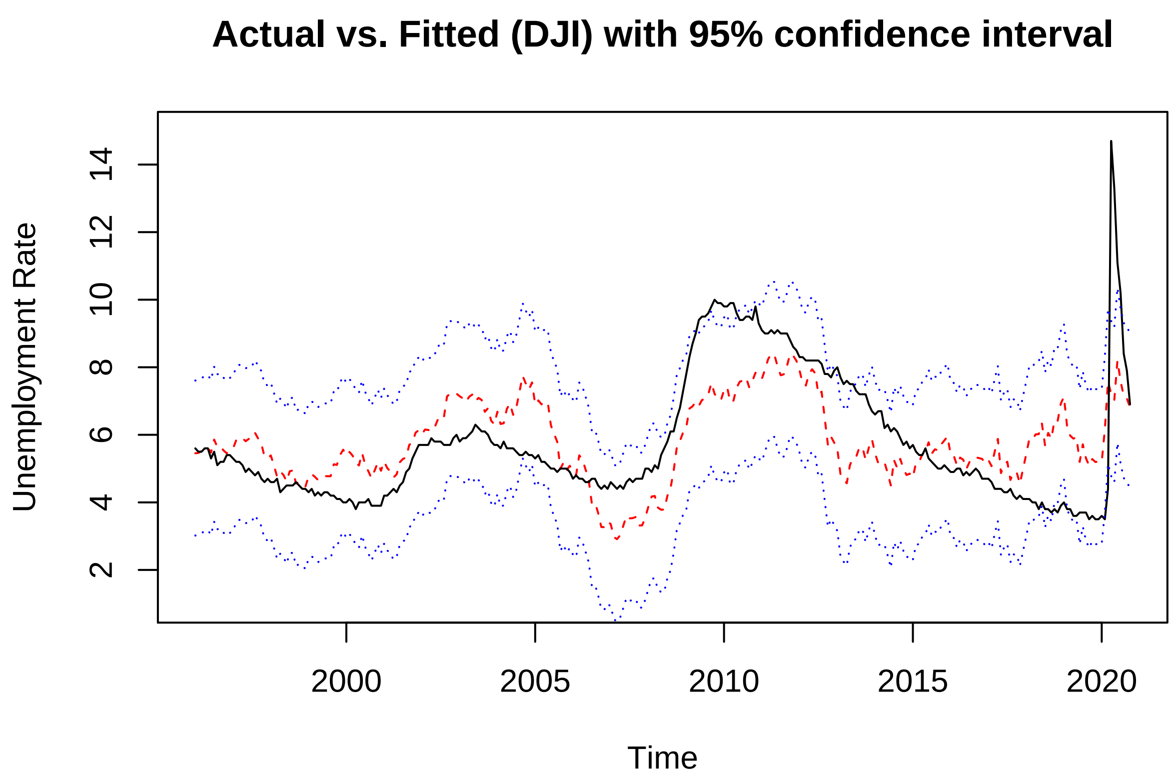
```r
e = m3.ex$fitted.values - window(data.dju, start = c(1992,1))
m = quantile(e, prob = c(0.05, 0.95))

plot(m3.ex$fitted.values, col = 'red', ylim = c(3,15), ylab = "Unemployment Rate", main = "Actual vs. F
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "start" is not a
## graphical parameter
```




```
## integer(0)
```

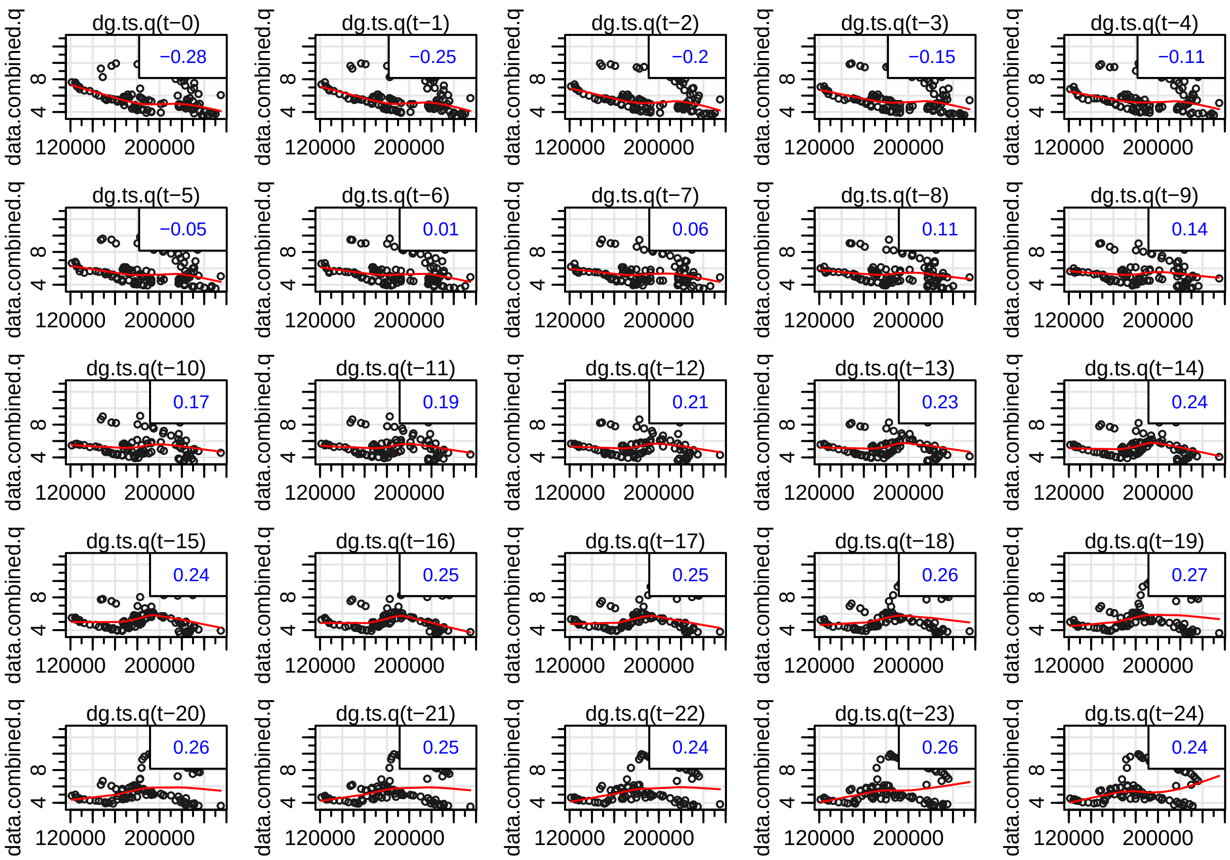
```r
plot(m3.ex$fitted.values, ylim = c(1, 15), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. I
lines( m3.ex$fitted.values + m[1], col = 'blue', lty = 3 ) +
lines( m3.ex$fitted.values + m[2], col = 'blue', lty = 3 )
```



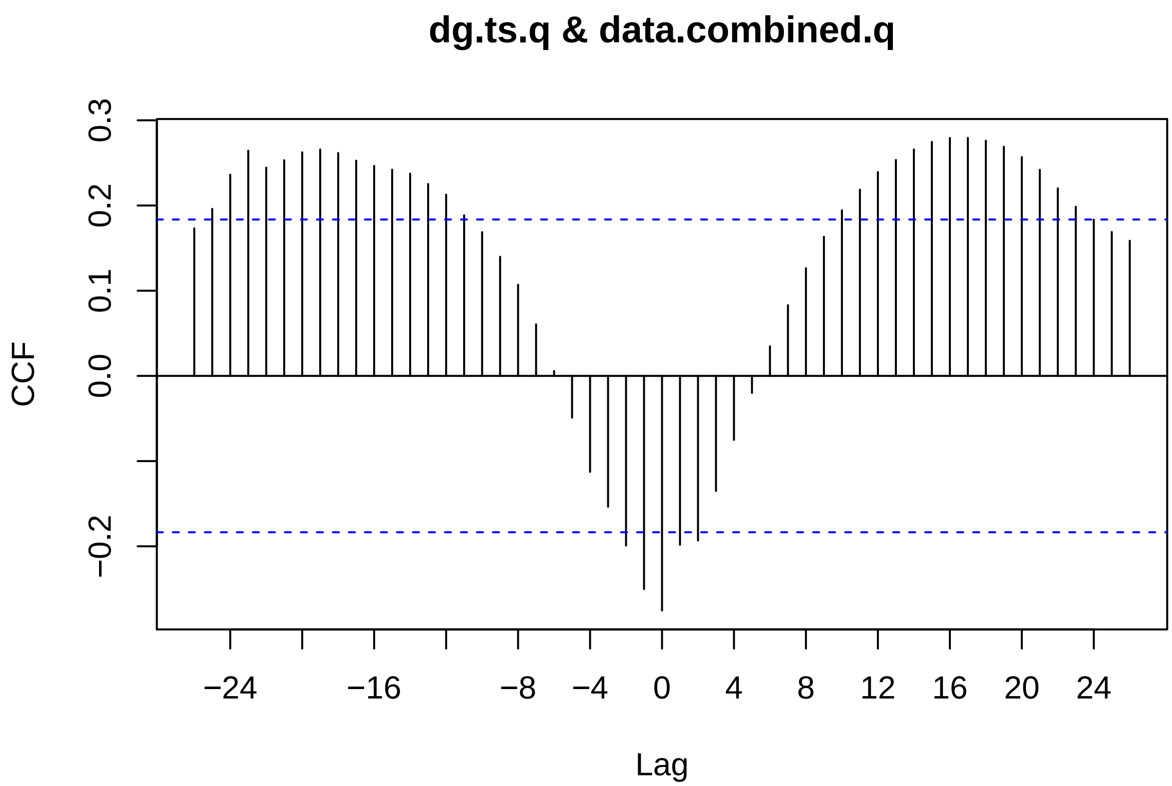
```
## integer(0)
```

Use all external variables in the same model

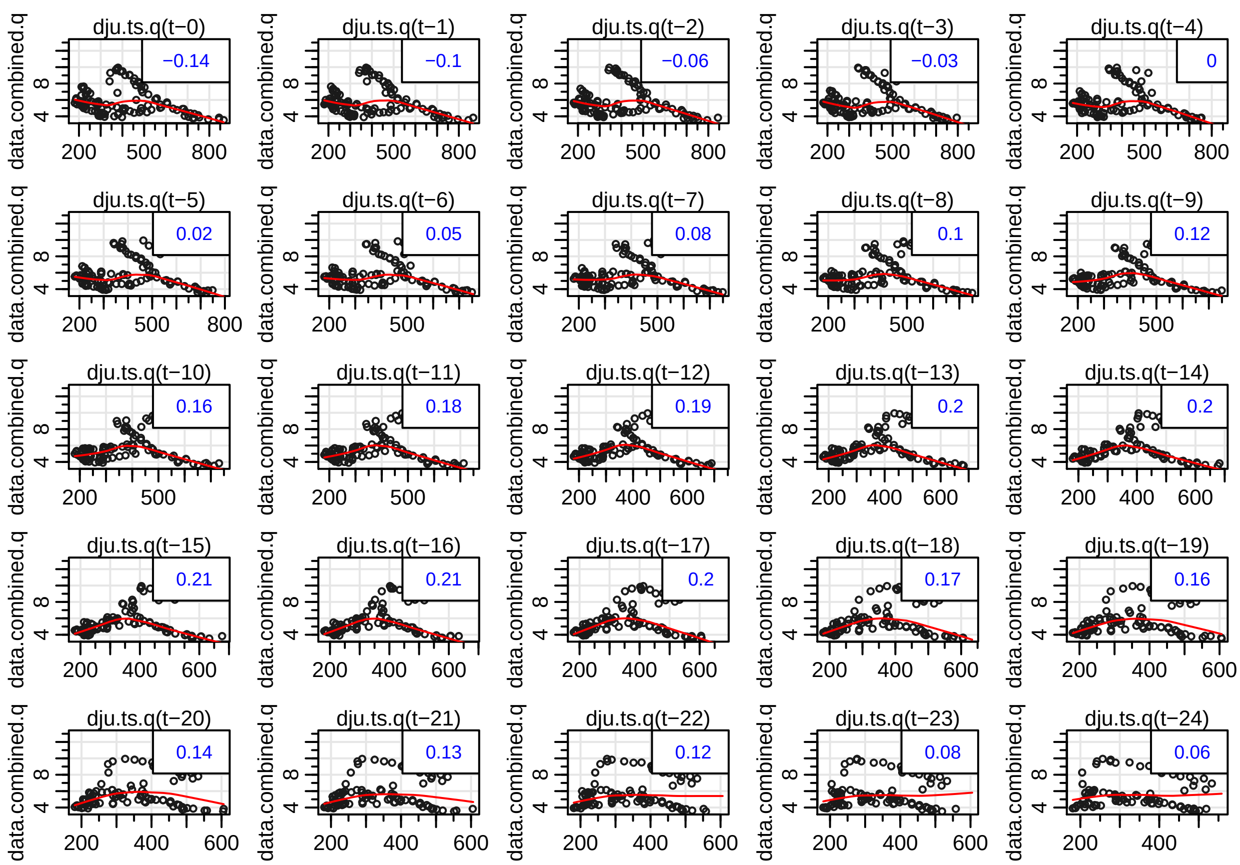
```
#Aggregate DGO and DJU to quarterly data

dg.ts.q = aggregate(window(dg.ts, start = c(1992, 4)), nfrequency = 4, FUN = 'mean')
dju.ts.q = aggregate(window(dju.ts, start = c(1992, 4)), nfrequency = 4, FUN = 'mean')

lag2.plot(dg.ts.q, data.combined.q, 24)
```



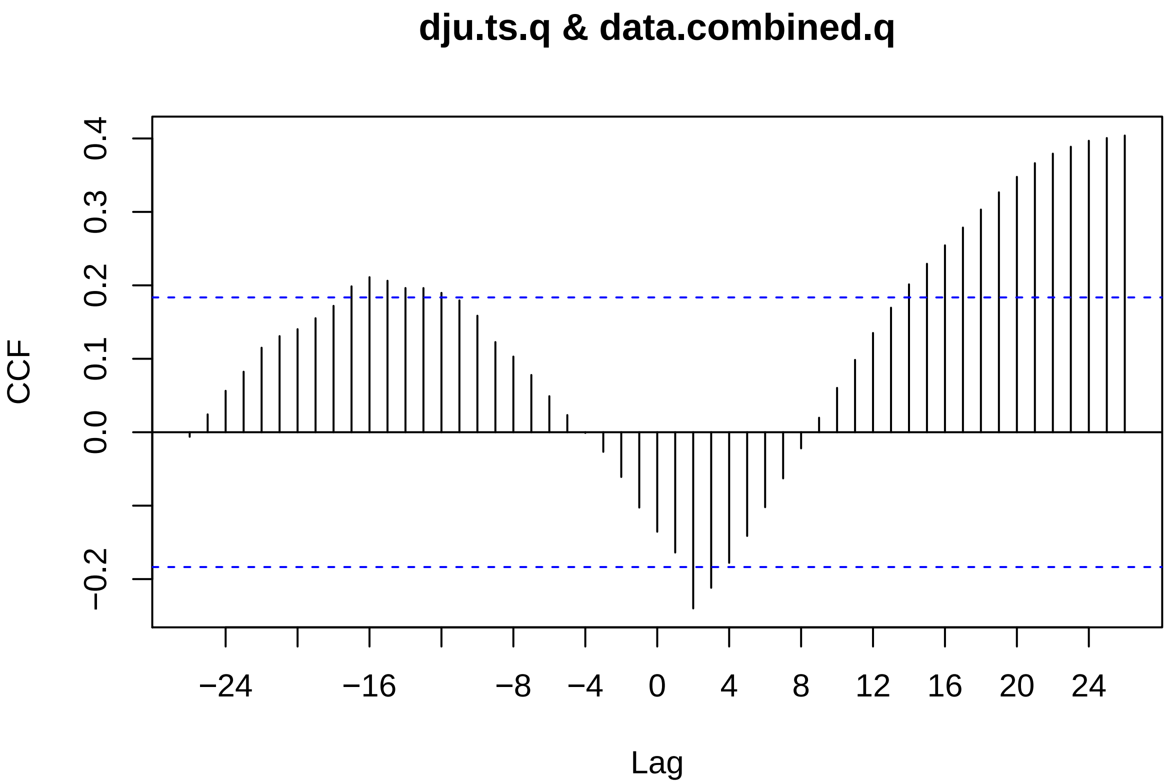

```
Ccf(dg.ts.q, data.combined.q, 26)
```




```r
lag2.plot(dju.ts.q, data.combined.q, 24)
```



```r
Ccf(dju.ts.q, data.combined.q, 26)
```




```r
newdata_4 = ts.intersect( UR = data.combined.q, Pce = pce.dt, lagPce10 = lag(pce.dt, -10), lagUR52 = la
head(newdata)
```

```
##                UR      Pce lagPce10  lagUR52
## 1961 Q3 6.766667 2053.774 1923.675 3.733333
## 1961 Q4 6.200000 2095.084 1953.384 3.666667
## 1962 Q1 5.633333 2117.277 1973.791 3.766667
## 1962 Q2 5.533333 2143.306 1976.014 3.833333
## 1962 Q3 5.566667 2160.580 1994.918 4.666667
## 1962 Q4 5.533333 2191.150 2020.082 5.866667
```

```r
m4.ex = tslm(UR ~ Pce + lagPce10 + lagUR52 + Dg + LagDg19 + Dju + lagDju16, data = newdata_4 )
summary(m4.ex)
```

```
##
## Call:
## tslm(formula = UR ~ Pce + lagPce10 + lagUR52 + Dg + LagDg19 +
##     Dju + lagDju16, data = newdata_4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0622 -0.2922  0.1082  0.4029  1.2713
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.624e+00  1.414e+00   6.101 3.22e-08 ***
## Pce         -4.688e-03  3.554e-04 -13.191  < 2e-16 ***
## lagPce10     4.510e-03  2.855e-04  15.797  < 2e-16 ***
```

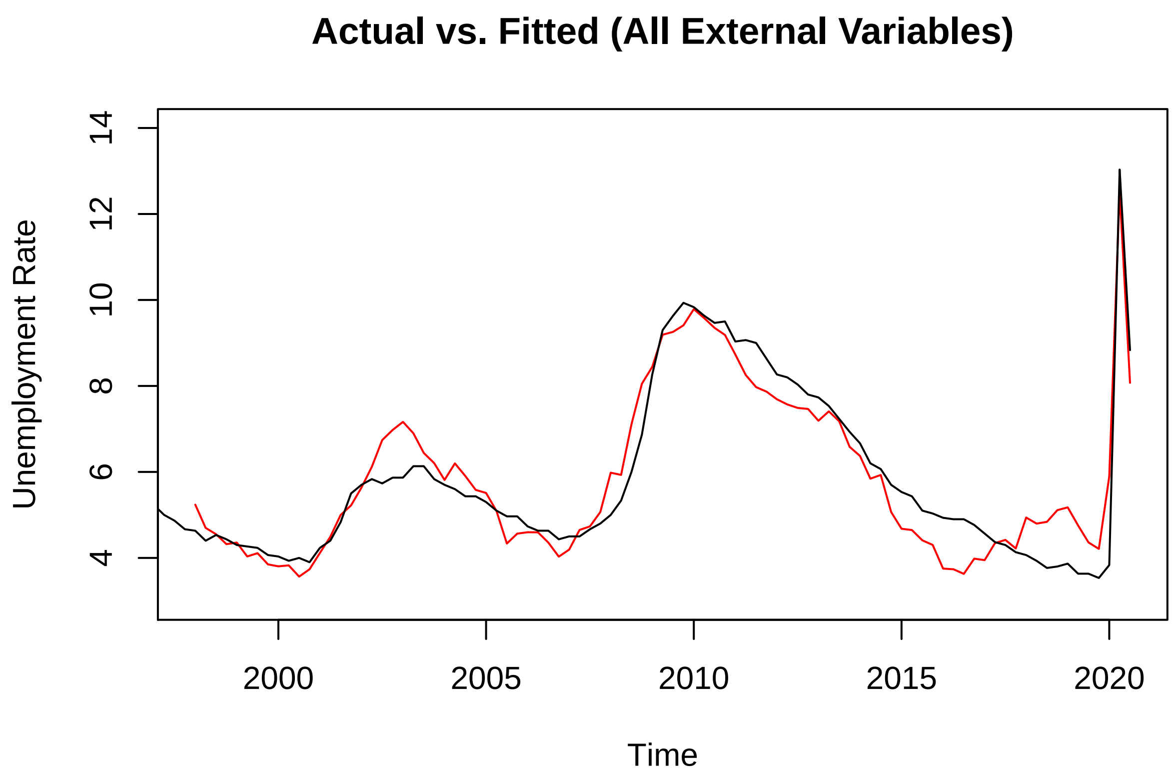
```
## lagUR52      -1.657e-01  1.018e-01  -1.628 0.107309
## Dg            8.400e-06  5.392e-06   1.558 0.123098
## LagDg19       1.009e-05  5.181e-06   1.948 0.054789 .
## Dju          -4.706e-03  1.232e-03  -3.819 0.000258 ***
## lagDju16      3.888e-03  1.589e-03   2.448 0.016486 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.675 on 83 degrees of freedom
## Multiple R-squared:  0.8897, Adjusted R-squared:  0.8804
## F-statistic: 95.66 on 7 and 83 DF,  p-value: < 2.2e-16
```

```r
accuracy( m4.ex$fitted.values, window(data.combined.q, start = c(1998,1)) )
```

```
##                        ME      RMSE       MAE       MPE     MAPE      ACF1
## Test set -7.318246e-17 0.6446047 0.4986046 -1.191987 9.591038 0.7594662
##          Theil's U
## Test set 0.5259039
```

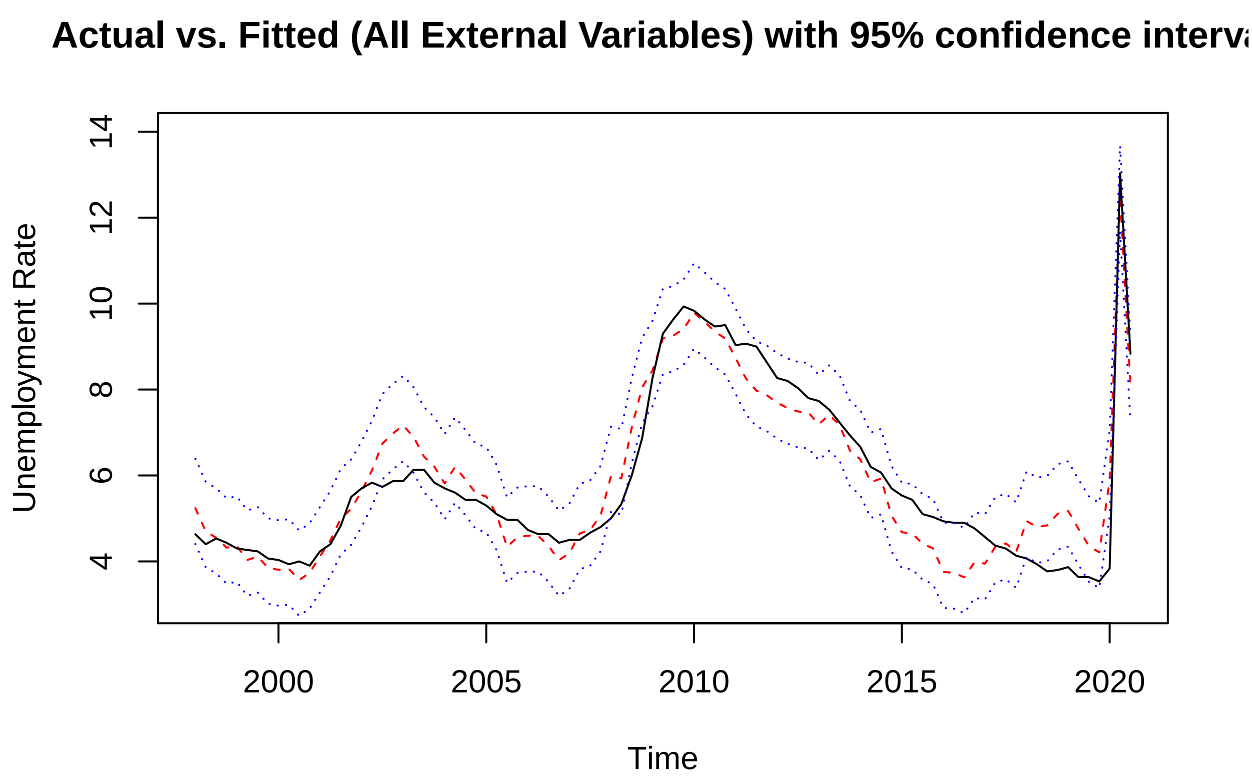
```r
e = m4.ex$fitted.values - window(data.combined.q, start = c(1998,1))
m = quantile(e, prob = c(0.05, 0.95))

plot(m4.ex$fitted.values, ylim = c(3, 14), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. F
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "start" is not a
## graphical parameter
```




```
## integer(0)
```

```r
plot(m4.ex$fitted.values, ylim = c(3, 14), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. F
lines( m4.ex$fitted.values + m[1], col = 'blue', lty = 3 ) +
lines( m4.ex$fitted.values + m[2], col = 'blue', lty = 3 )
```




```
## integer(0)
```

Use Only Lagged Variables

```
m5.ex = tslm(UR ~  lagPce10 + lagUR52 + LagDg19 + LagDg23 + lagDju16, data = newdata_4 )
summary(m5.ex)
```

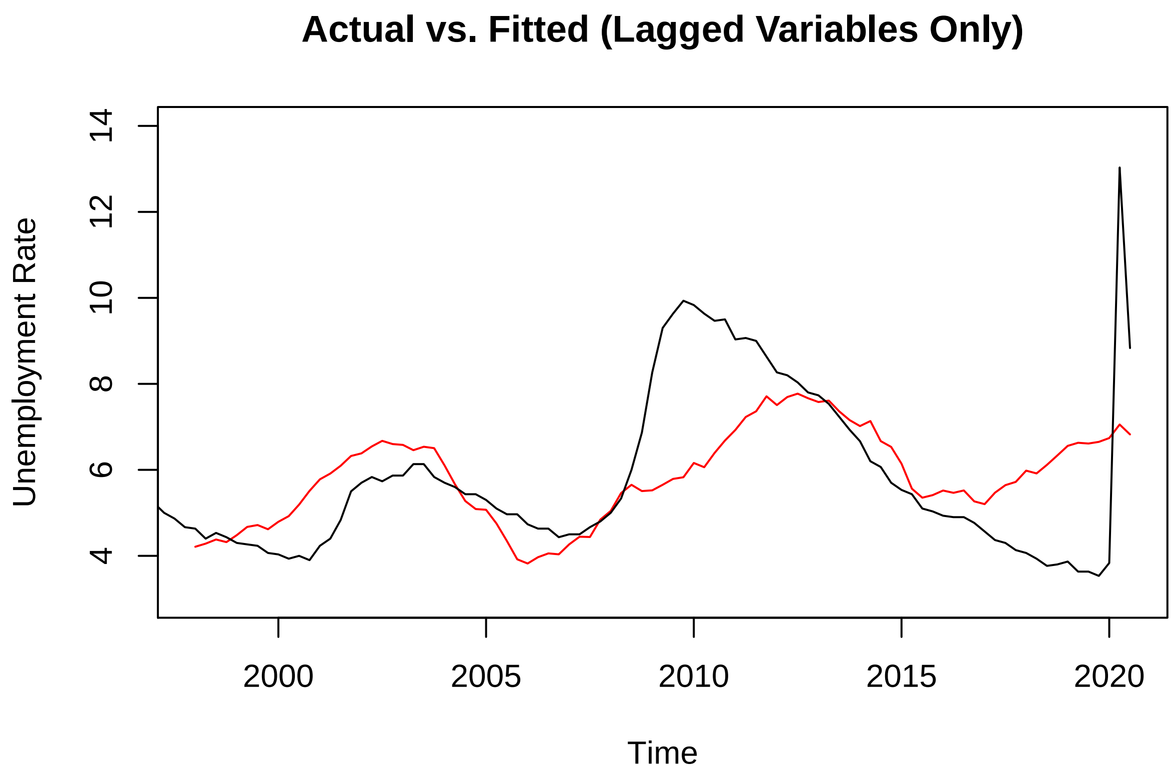
```
##
## Call:
## tslm(formula = UR ~ lagPce10 + lagUR52 + LagDg19 + LagDg23 +
##     lagDju16, data = newdata_4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1175 -0.7461 -0.1814  0.4101  5.9795
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.295e+01  2.890e+00   4.480 2.31e-05 ***
## lagPce10    -4.320e-04  2.622e-04  -1.648    0.103
## lagUR52     -1.050e+00  2.363e-01  -4.445 2.64e-05 ***
## LagDg19      9.412e-06  1.255e-05   0.750    0.455
## LagDg23      4.445e-06  1.083e-05   0.411    0.682
## lagDju16     1.495e-03  3.628e-03   0.412    0.681
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.694 on 85 degrees of freedom
## Multiple R-squared:  0.2887, Adjusted R-squared:  0.2468
## F-statistic: 6.899 on 5 and 85 DF,  p-value: 1.897e-05
```

```r
accuracy( m5.ex$fitted.values, window(data.combined.q, start = c(1998,1)) )
```

```
##                     ME     RMSE      MAE       MPE    MAPE     ACF1 Theil's U
## Test set -1.75779e-16 1.637135 1.145435 -6.663878 20.2306 0.777331  1.267012
```

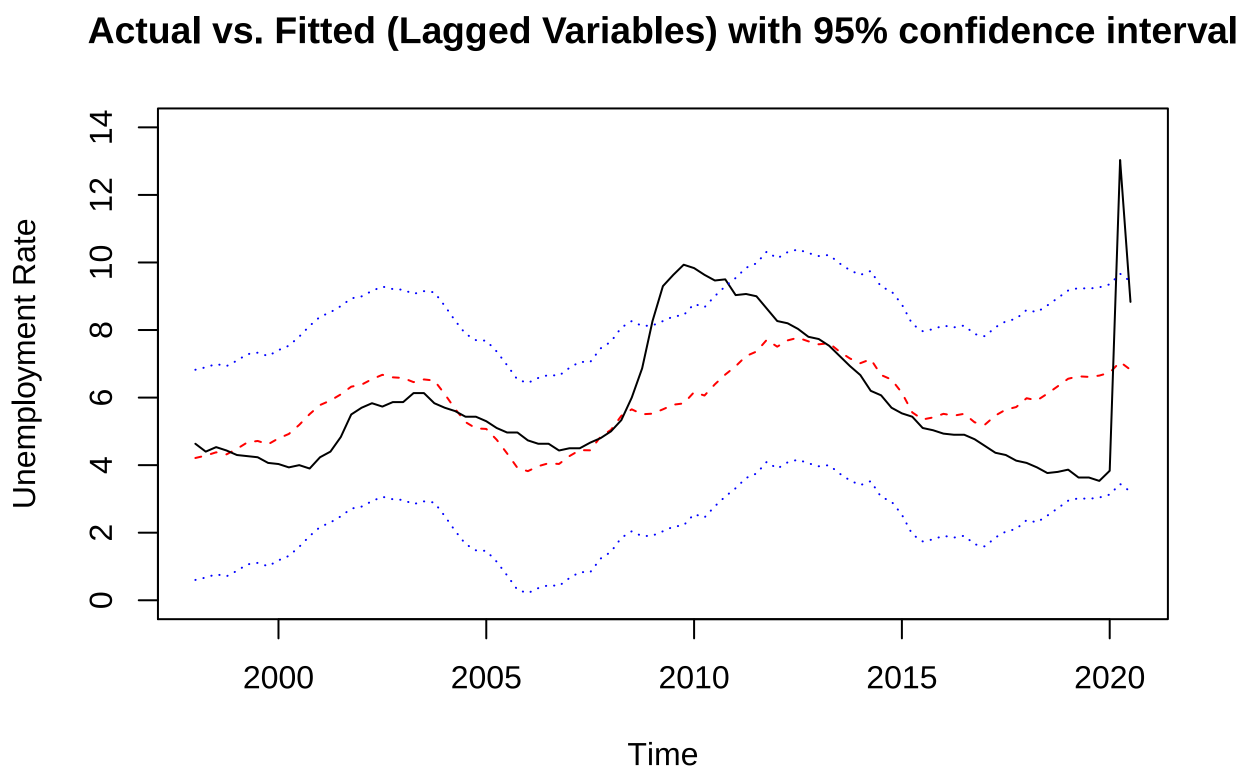
```r
plot(m5.ex$fitted.values, ylim = c(3, 14), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. F
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "start" is not a
## graphical parameter
```




```
## integer(0)
```

```r
e = m5.ex$fitted.values - window(data.combined.q, start = c(1998,1))
m = quantile(e, prob = c(0.05, 0.95))

plot(m5.ex$fitted.values, ylim = c(0, 14), col = 'red', ylab = "Unemployment Rate", main = "Actual vs. F
lines( m5.ex$fitted.values + m[1], col = 'blue', lty = 3 ) +
lines( m5.ex$fitted.values + m[2], col = 'blue', lty = 3 )
```




```
## integer(0)
```

Using only lagged variables, the model is able to explain 24.68% of the variation of Unemployment. Though this is not very high, it is still very valuable because the model is able to predict the changes in trends of the unemployment rates.