1. Load libraries

```
library(tseries)
library(fBasics)
```

```
## Loading required package: timeDate
## Loading required package: timeSeries
##
## Rmetrics Package fBasics
## Analysing Markets and calculating Basic Statistics
## Copyright (C) 2005-2014 Rmetrics Association Zurich
## Educational Software for Financial Engineering and Computational Science
## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.
## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

```
library(forecast)
library(lmtest)
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following object is masked from 'package:timeSeries':
##
##     time<-
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

2. Import Data

```
setwd("~/Desktop/CSC425/week6/indpro_sarima")
myd = read.table("INDPRO_nsa.csv", header = T, sep = ',')
head(myd)
```

```
##       date  indpro
## 1 1/1/1960 23.7741
## 2 2/1/1960 23.9082
## 3 3/1/1960 23.8546
## 4 4/1/1960 23.6667
## 5 5/1/1960 23.6131
## 6 6/1/1960 23.6667
```

```
tail(myd)
```

```
##          date   indpro
## 644  8/1/2013 101.6590
## 645  9/1/2013 101.3336
## 646 10/1/2013 100.4917
```

```
## 647 11/1/2013 100.0904
## 648 12/1/2013 101.0132
## 649  1/1/2014 100.9218
```

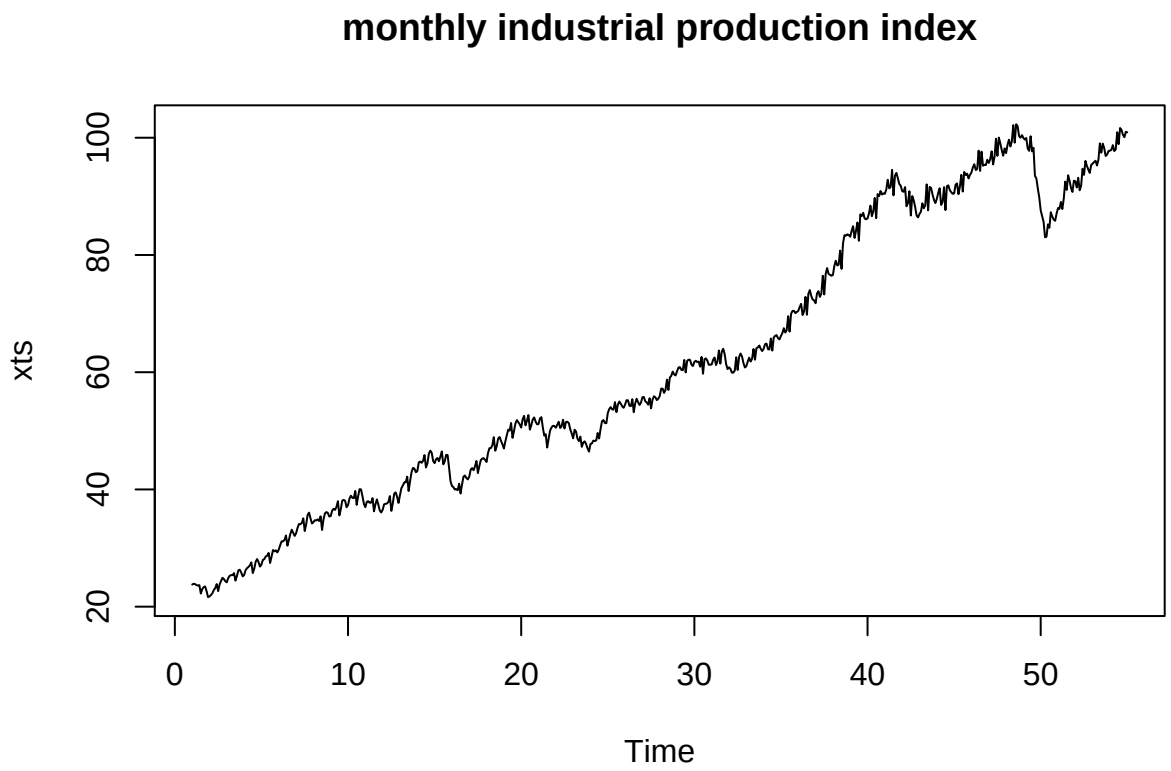
creates time series object

```
x = myd$indpro
xts = ts(x,frequency = 12, start(1960,1))
head(xts,20)
```

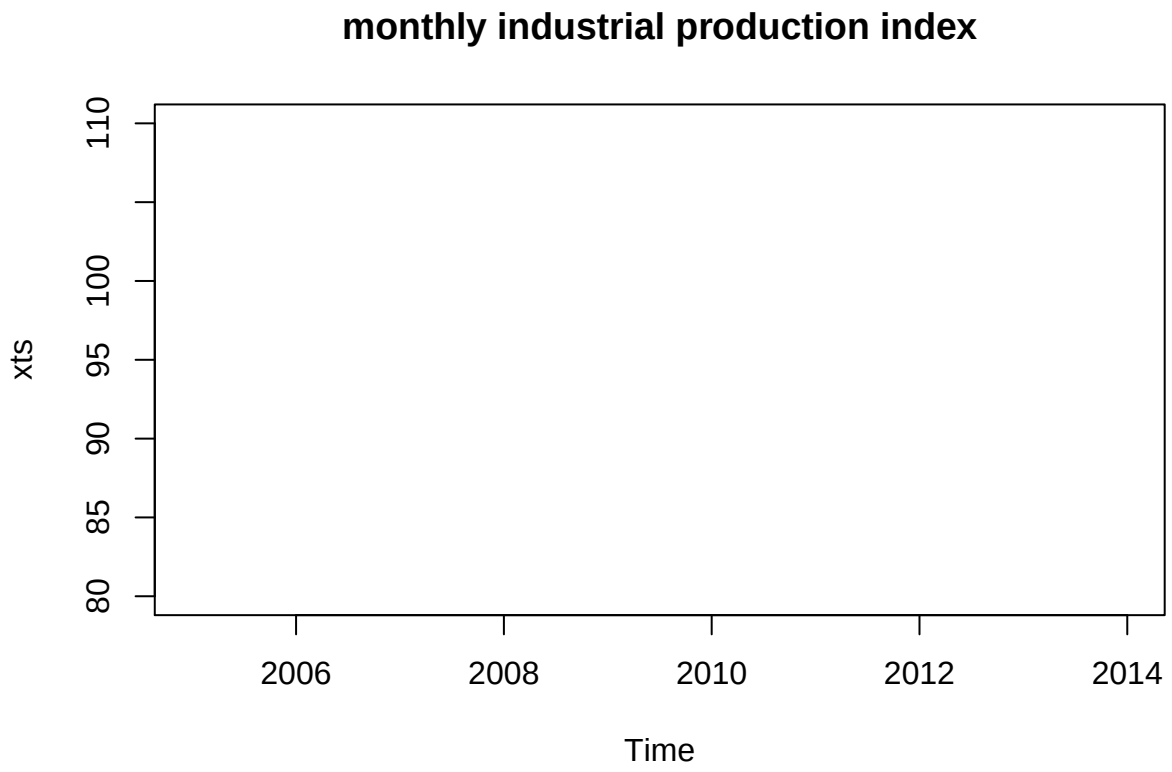
```
##       Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep
## 1 23.7741 23.9082 23.8546 23.6667 23.6131 23.6667 22.2446 22.9959 23.3447
## 2 21.7348 21.9762 22.2714 22.8349 23.1837 23.8546 22.6471 23.8009
##       Oct     Nov     Dec
## 1 23.4521 22.6202 21.6274
## 2
```

create time plot

```
plot(xts, main = "monthly industrial production index")
```




```
plot(xts, main = "monthly industrial production index", xlim = c(2005, 2014), ylim=c(80,110))
```




```
hist(x, main = "Monthly industiral production index")
```

3. ACF Analysis

two plots per page

```
par(mfcol = c(1,1))
```

Acf plot for log starts

as.vector() transforms time series to regurlar numerical vector

```
acf(as.vector(x), lab.max = 24, main = "ACF of log starts")
```

```
## Warning in plot.window(...): "lab.max" is not a graphical parameter
```

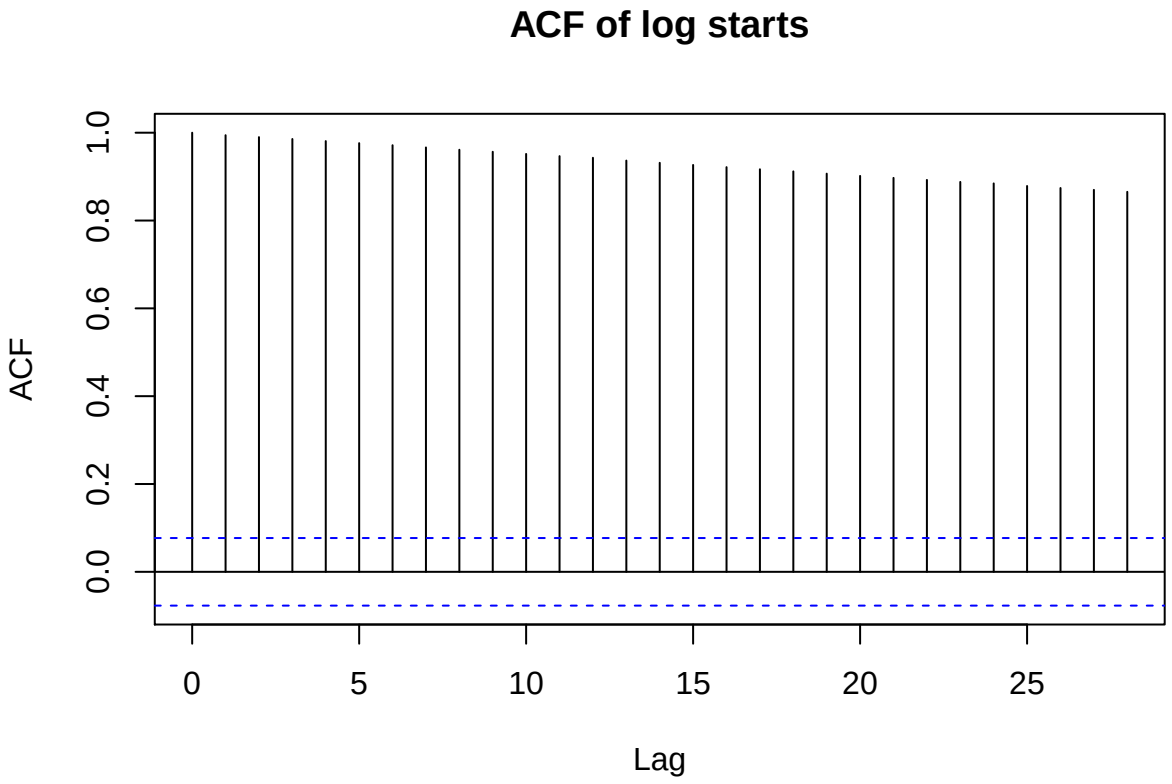
```
## Warning in plot.xy(xy, type, ...): "lab.max" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "lab.max" is
## not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "lab.max" is
## not a graphical parameter
```

```
## Warning in box(...): "lab.max" is not a graphical parameter
```

```
## Warning in title(...): "lab.max" is not a graphical parameter
```



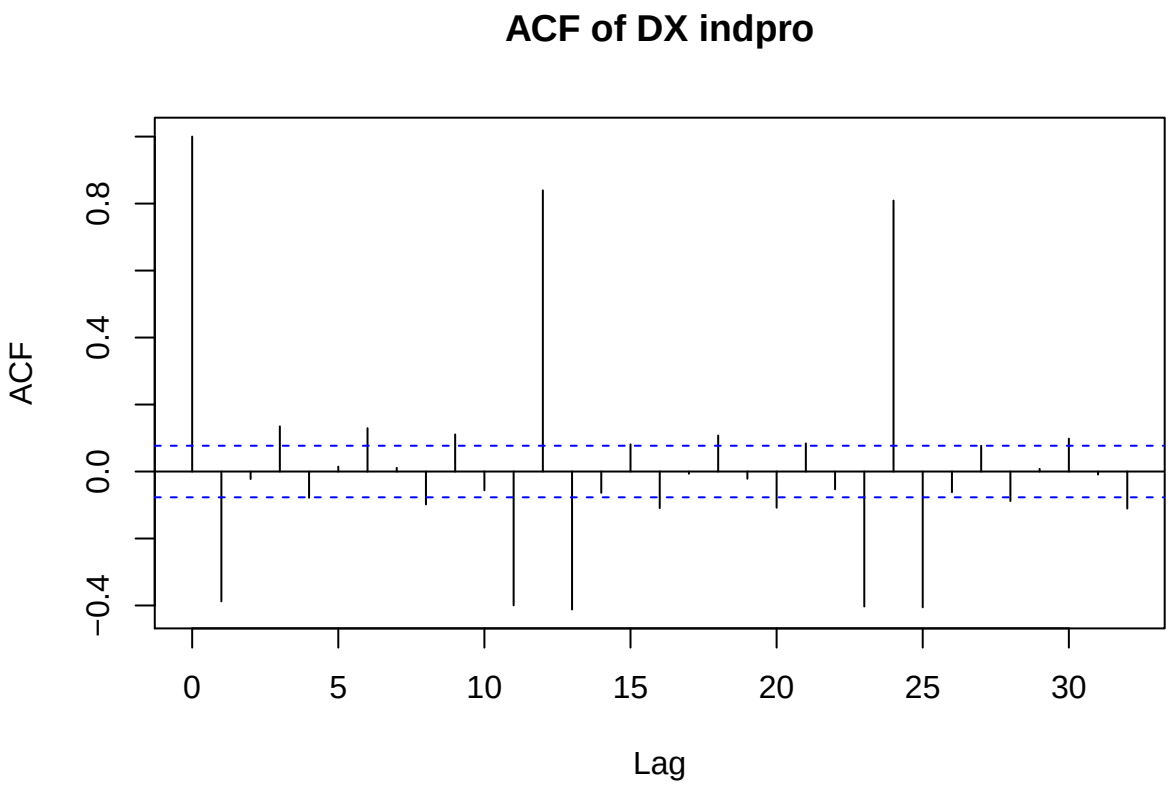
4. Applying differncing to data

compute regular differences

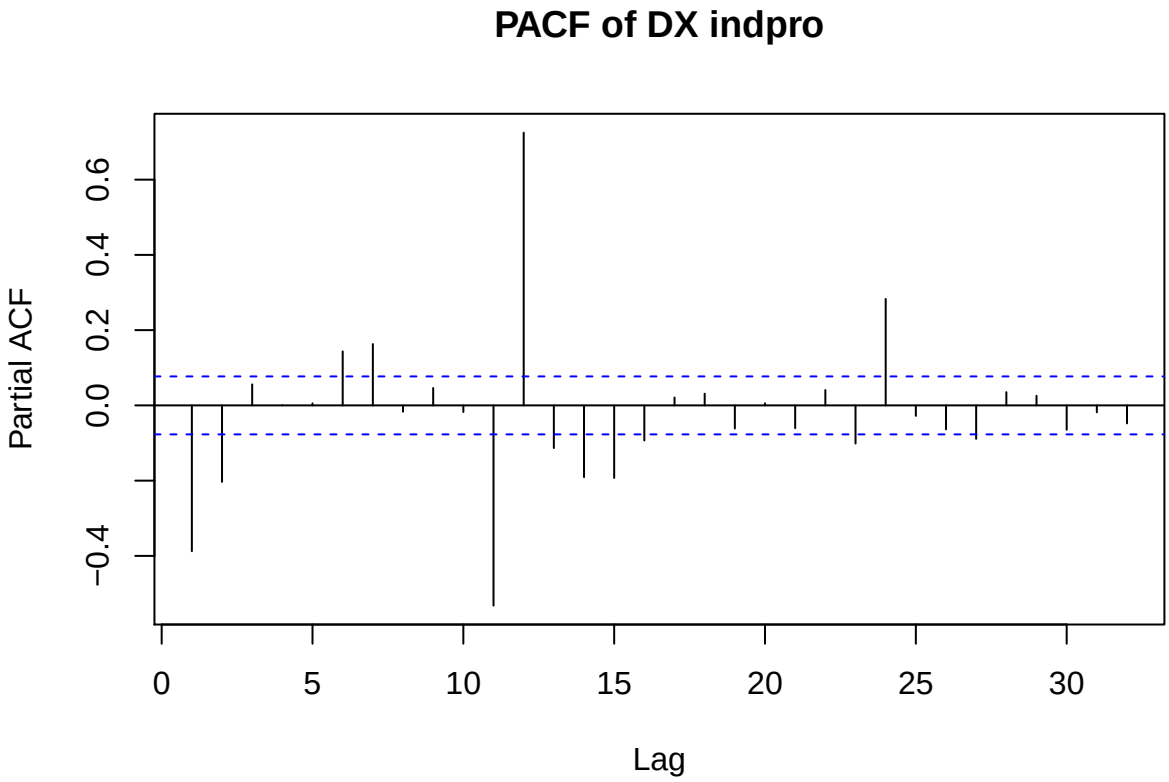
```
dx = diff(x)
```

create acf plot

```
acf(as.vector(dx), lag.max = 32, main = "ACF of DX indpro")
```




```
pacf(as.vector(dx), lag.max = 32, main = "PACF of DX indpro")
```



5. Fit SARIMA model

```
library(forecast)
```

Try automated order selection

since TS is stationary, stationary = FALSE;

```
auto.arima(xts,trace = T, seasonal = TRUE)
```

```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,1,2)(1,0,1)[12] with drift         : Inf
##  ARIMA(0,1,0)            with drift         : 2211.63
##  ARIMA(1,1,0)(1,0,0)[12] with drift         : 1385.786
##  ARIMA(0,1,1)(0,0,1)[12] with drift         : 1787.676
##  ARIMA(0,1,0)                               : 2214.792
##  ARIMA(1,1,0)            with drift         : 2109.143
##  ARIMA(1,1,0)(2,0,0)[12] with drift         : Inf
##  ARIMA(1,1,0)(1,0,1)[12] with drift         : Inf
##  ARIMA(1,1,0)(2,0,1)[12] with drift         : Inf
##  ARIMA(0,1,0)(1,0,0)[12] with drift         : 1391.893
##  ARIMA(2,1,0)(1,0,0)[12] with drift         : 1356.245
##  ARIMA(2,1,1)(1,0,0)[12] with drift         : 1332.642
##  ARIMA(3,1,2)(1,0,0)[12] with drift         : 1327.221
```

```
##  ARIMA(3,1,2)(1,0,0)[12]                    : 1325.294
##  ARIMA(3,1,2)                               : 2087.701
##  ARIMA(3,1,2)(2,0,0)[12]                    : Inf
##  ARIMA(3,1,2)(1,0,1)[12]                    : Inf
##  ARIMA(3,1,2)(2,0,1)[12]                    : Inf
##  ARIMA(2,1,2)(1,0,0)[12]                    : 1324.546
##  ARIMA(2,1,1)(1,0,0)[12]                    : 1330.725
##  ARIMA(2,1,3)(1,0,0)[12]                    : Inf
##  ARIMA(1,1,1)(1,0,0)[12]                    : Inf
##  ARIMA(3,1,3)(1,0,0)[12]                    : Inf
##  ARIMA(2,1,2)(1,0,0)[12] with drift         : 1326.461
##  ARIMA(2,1,2)                               : 2099.465
##  ARIMA(2,1,2)(2,0,0)[12]                    : Inf
##  ARIMA(2,1,2)(1,0,1)[12]                    : Inf
##  ARIMA(2,1,2)(2,0,1)[12]                    : Inf
##  ARIMA(1,1,2)(1,0,0)[12]                    : 1324.152
##  ARIMA(1,1,3)(1,0,0)[12]                    : 1323.852
##  ARIMA(0,1,2)(1,0,0)[12]                    : 1366.236
##  ARIMA(2,1,4)(1,0,0)[12]                    : Inf
##  ARIMA(1,1,3)(1,0,0)[12] with drift         : 1325.725
##  ARIMA(1,1,3)                               : 2085.006
##  ARIMA(1,1,3)(2,0,0)[12]                    : Inf
##  ARIMA(1,1,3)(1,0,1)[12]                    : Inf
##  ARIMA(1,1,3)(2,0,1)[12]                    : Inf
##  ARIMA(0,1,3)(1,0,0)[12]                    : 1335.901
##  ARIMA(1,1,4)(1,0,0)[12]                    : 1325.122
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(1,1,3)(1,0,0)[12]                    : 1332.072
##
##  Best model: ARIMA(1,1,3)(1,0,0)[12]
```

```
## Series: xts
## ARIMA(1,1,3)(1,0,0)[12]
##
## Coefficients:
##          ar1      ma1     ma2     ma3    sar1
##       0.7107  -0.6793  0.1675  0.0660  0.8675
## s.e.  0.0934   0.1031  0.0494  0.0441  0.0187
##
## sigma^2 estimated as 0.4406:  log likelihood=-659.97
## AIC=1331.94   AICc=1332.07   BIC=1358.78
```

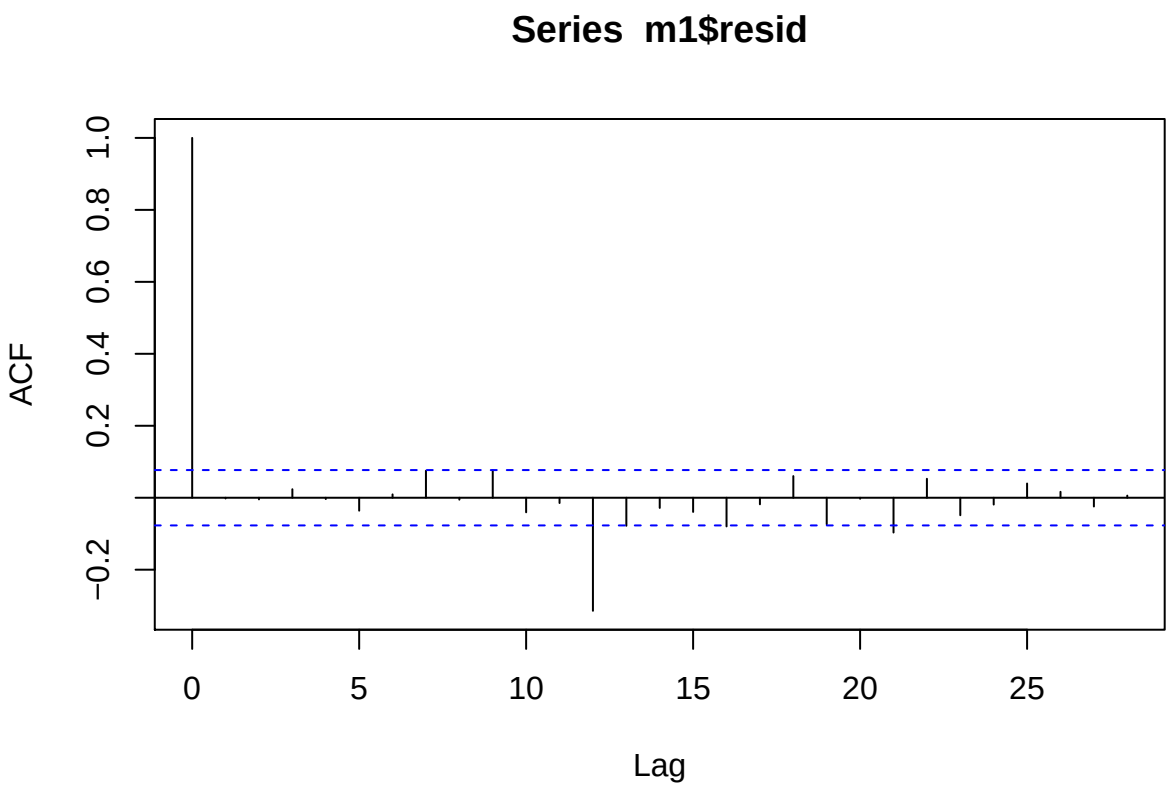
fit multiplicative seasonal models ARIMA(1,1,3)(1,0,0)[12] selected by BIC criterion

```
m1 = Arima(x, order = c(1,1,3), seasonal = list(order = c(1,0,0), period = 12), method = "ML")
coeftest(m1)
```

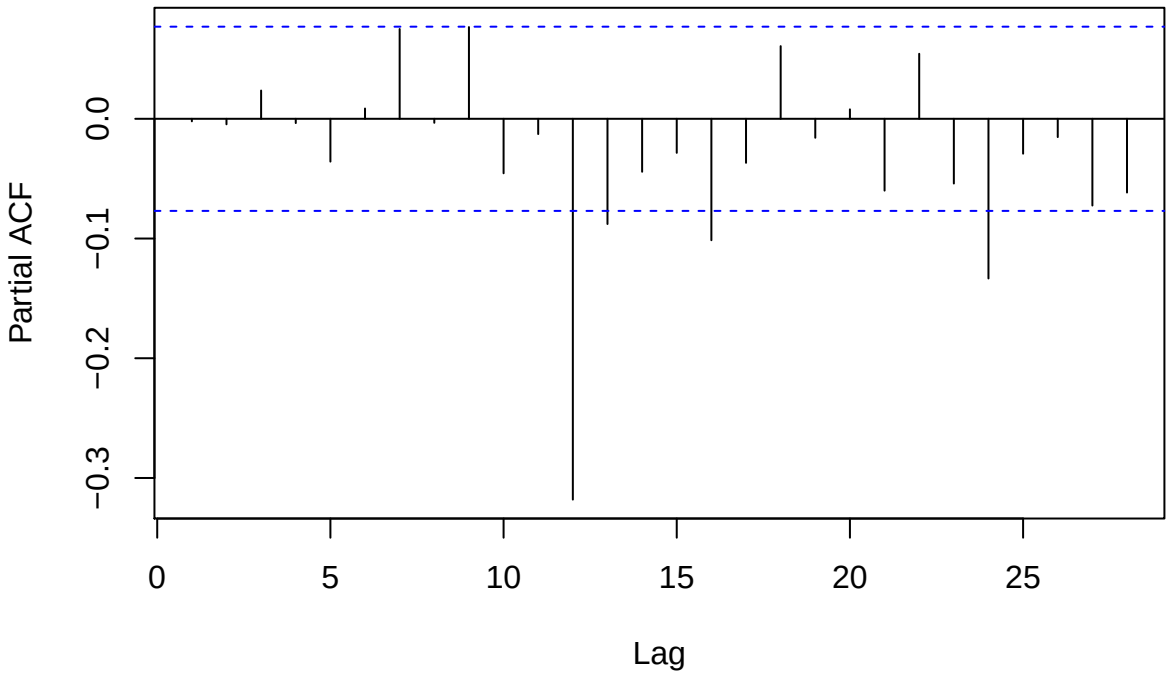
```
##
## z test of coefficients:
##
##       Estimate Std. Error z value  Pr(>|z|)
## ar1   0.710755   0.093337  7.6149 2.638e-14 ***
## ma1  -0.679269   0.103014 -6.5940 4.283e-11 ***
```

```
## ma2   0.167343   0.049372  3.3894 0.0007004 ***
## ma3   0.065979   0.044077  1.4969 0.1344195
## sar1  0.867501   0.018735 46.3038 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
acf(m1$resid)
```




```
pacf(m1$resid)
```

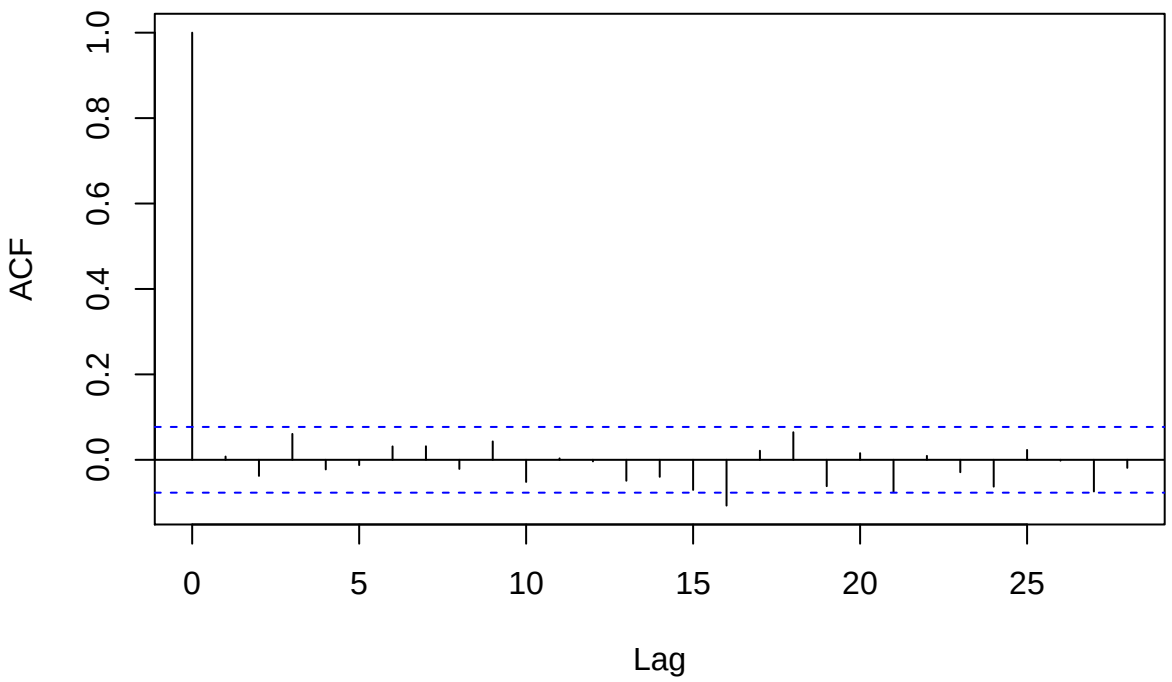


revise model M1 and fit model M2 ARIMA(1,1,2)(0,1,1)[12]

```
m2 = Arima(x, order = c(1,1,2), seasonal = list(order = c(0,1,1), period = 12), method = "ML")
coeftest(m2)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value  Pr(>|z|)
## ar1   0.807520   0.052032  15.5196 < 2.2e-16 ***
## ma1  -0.770623   0.062347 -12.3602 < 2.2e-16 ***
## ma2   0.164537   0.042385   3.8819 0.0001036 ***
## sma1 -0.625494   0.031393 -19.9245 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
acf(m2$resid)
```



"approximated" Ljung box tests on residuals

```
Box.test(m2$residuals, 7, "Ljung-Box",fitdf = length(m2$coef))
```

```
##
##  Box-Ljung test
##
## data:  m2$residuals
## X-squared = 5.1021, df = 3, p-value = 0.1645
```

```
Box.test(m2$residuals, 10, "Ljung-Box",fitdf = length(m2$coef))
```

```
##
##  Box-Ljung test
##
## data:  m2$residuals
## X-squared = 8.3959, df = 6, p-value = 0.2105
```

```
Box.test(m2$residuals, 15, "Ljung-Box",fitdf = length(m2$coef))
```

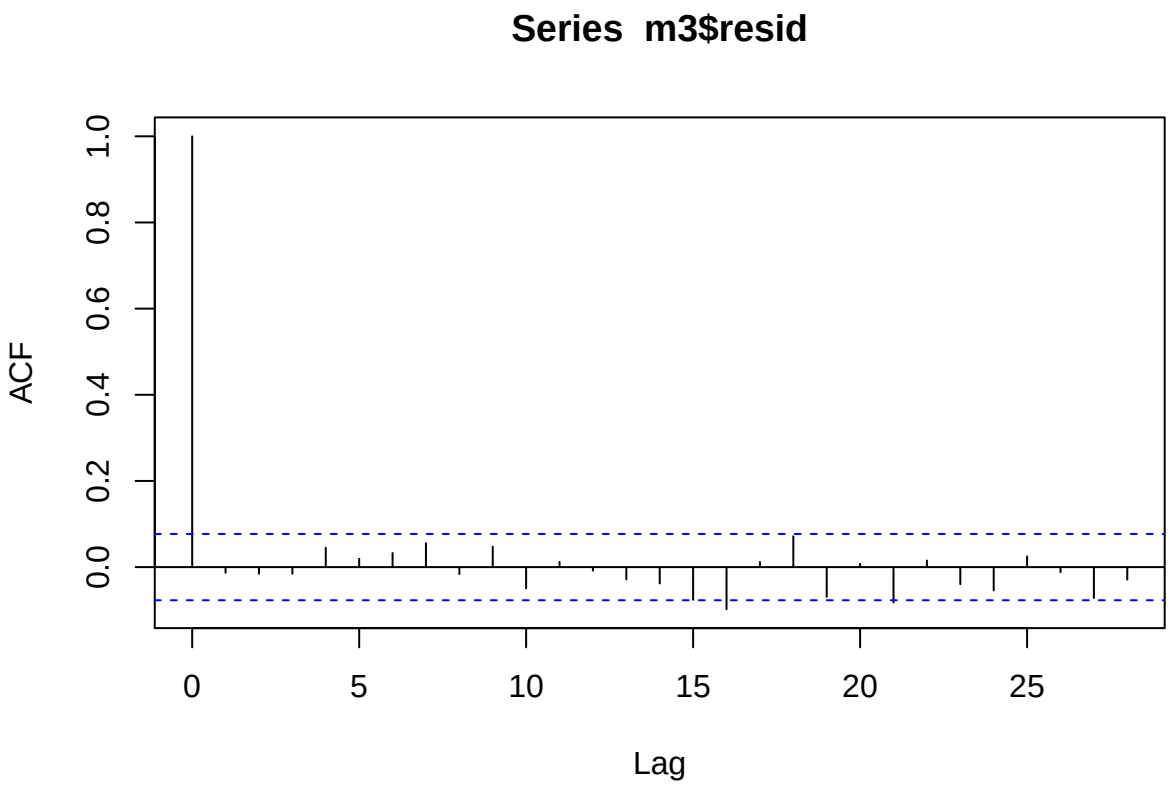
```
##
##  Box-Ljung test
##
## data:  m2$residuals
## X-squared = 14.369, df = 11, p-value = 0.2132
```

New initial model with lag AR(3) order and Seasonal MA(1) order

```
m3 = Arima(x, order = c(3,1,0), seasonal = list(order=c(0,1,1), period = 12), method = "ML")
coeftest(m3)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value  Pr(>|z|)
## ar1   0.064022   0.038748   1.6522   0.09849 .
## ar2   0.170996   0.038350   4.4588 8.241e-06 ***
## ar3   0.214525   0.038724   5.5399 3.027e-08 ***
## sma1 -0.622719   0.031214 -19.9502 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
acf(m3$resid)
```



"approximated" Ljung box test on residuals

```
Box.test(m3$residuals, 7, "Ljung-Box", fitdf = length(m3$coef))
```

```
##
##  Box-Ljung test
##
## data:  m3$residuals
## X-squared = 4.7028, df = 3, p-value = 0.1949
```

```
Box.test(m3$residuals, 10, "Ljung-Box", fitdf = length(m3$coef))
```

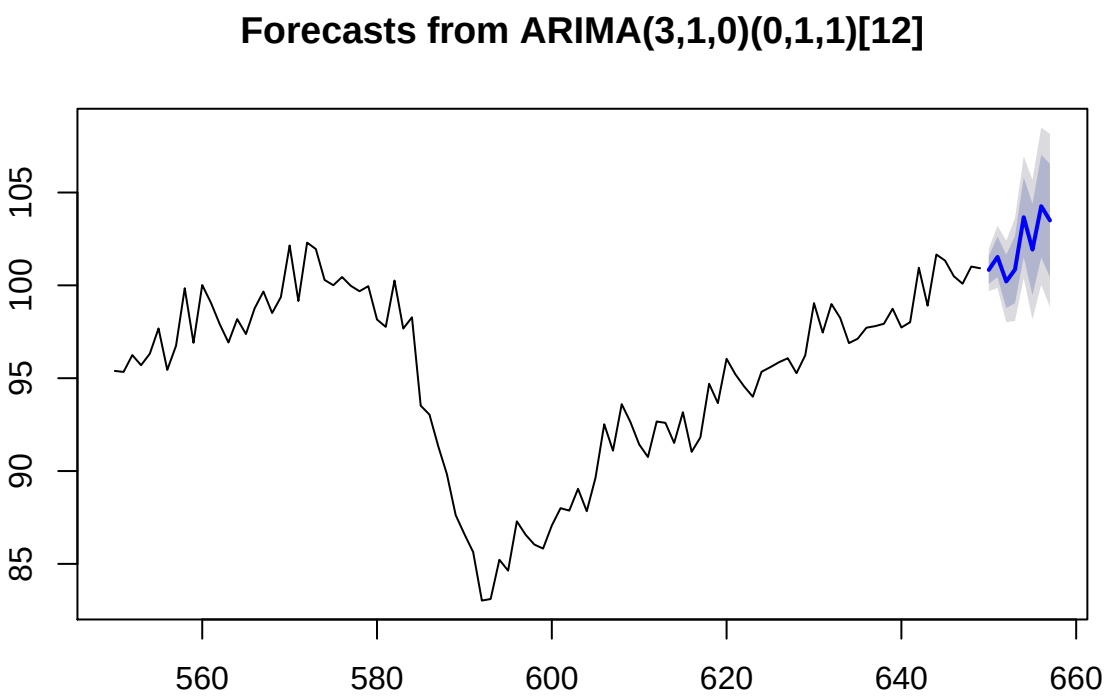
```
##
##  Box-Ljung test
##
## data:  m3$residuals
## X-squared = 7.9815, df = 6, p-value = 0.2395
```

compute predictions for log data for up to 8-step aheads

```
f1 = forecast(m3, h = 8)
f1
```

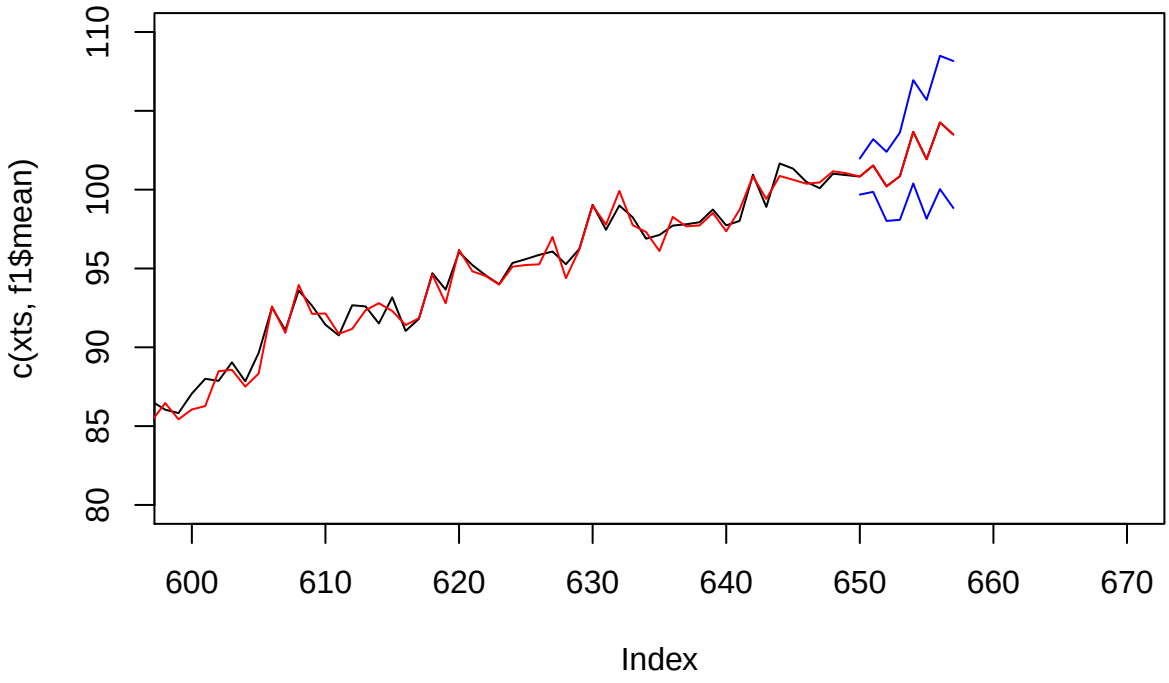
```
##     Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 650       100.8321 100.08337 101.5808  99.68703 101.9771
## 651       101.5310 100.43773 102.6242  99.85900 103.2029
## 652       100.2091  98.77525 101.6429  98.01624 102.4019
## 653       100.8523  99.04208 102.6624  98.08384 103.6207
## 654       103.6682 101.52417 105.8122 100.38919 106.9472
## 655       101.9253  99.46331 104.3873  98.16000 105.6906
## 656       104.2623 101.49680 107.0279 100.03280 108.4919
## 657       103.4972 100.44868 106.5458  98.83486 108.1596
```

```
plot(f1, include = 100)
```



forecast plot with prediction error

```
plot(c(xts, f1$mean), type = "l", xlim=c(600,670), ylim = c(80,110))
lines(c(f1$fitted, f1$mean), col = "red")
lines(c(rep(NA, length(f1$fitted)), f1$upper[,2]), col = "blue")
lines(c(rep(NA, length(f1$fitted)), f1$lower[,2]), col = "blue")
```



model validation using backtesting

```
source("backtest.R")
backtest(m3, x, 520, 1)
```

```
## [1] "RMSE of out-of-sample forecasts"
## [1] 0.832492
## [1] "Mean absolute error of out-of-sample forecasts"
## [1] 0.5982626
## [1] "Mean Absolute Percentage error"
## [1] 0.006380121
## [1] "Symmetric Mean Absolute Percentage error"
## [1] 0.006376156
```