Week 6: ARIMA (Part 2)

Xuan Pham

9/24/2019

Last week, we learned:

* ARIMA(p,d,0) is appropriate if: 1) ACF plot is exponentially decaying or sinusoidal and 2) PACF plot shows a significant spike at lag p but nothing else after.
* ARIMA(0,d,q) is appropriate if: 1) ACF plot shows a significant spike at lag q but nothing else after and 2) PACF plot is exponentially decaying or sinusoidal

# A New Case

What patterns do you see in the Quarterly Percentage Change in US Consumption Expenditure time series?

## Loading required package: ggplot2

## Loading required package: forecast

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

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

## Registered S3 methods overwritten by 'forecast':  
## method from   
## fitted.fracdiff fracdiff  
## residuals.fracdiff fracdiff

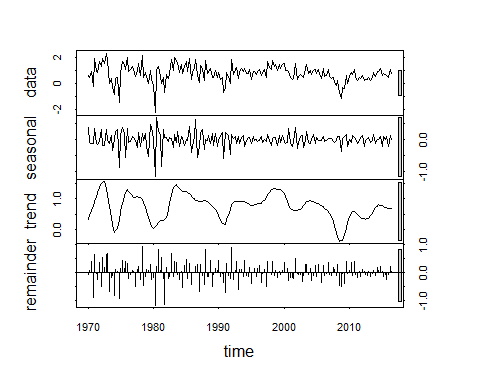
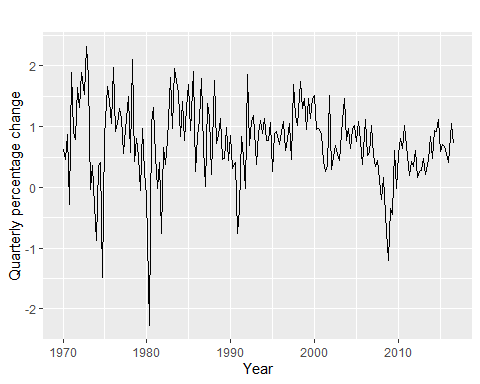
## Loading required package: fma

## Loading required package: expsmooth

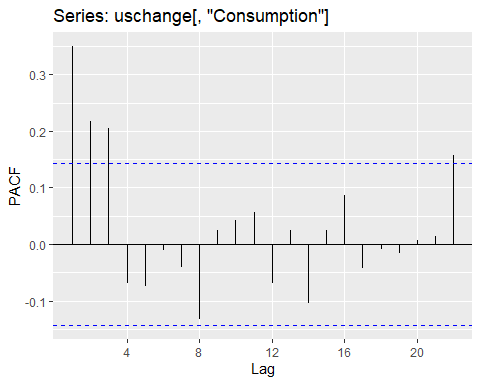
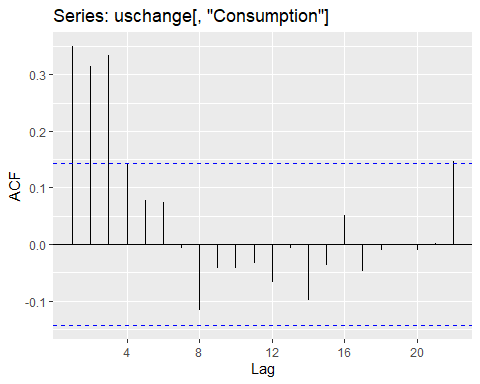
## -- Attaching packages ------------------------ tidyverse 1.2.1 --

## v tibble 2.1.3 v purrr 0.3.2  
## v tidyr 0.8.3 v dplyr 0.8.2  
## v readr 1.3.1 v stringr 1.4.0  
## v tibble 2.1.3 v forcats 0.4.0

## -- Conflicts --------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()



Let’s look at the serial correlation.



ACF plot shows 3 significant lags and perhaps even 4?  
PACF plot also shows 3 significant lags.

Now that we have positive p and q lags, what do we do?

We can account for the 3 lags in the ACF and have something like this: ARIMA(3,0,0). If we want to account for the 4th lag, we can do ARIMA(4,0,0).  
We can also account for the 3 lags in the PACF and have something like this: ARIMA(0,0,3).  
We can even have this: ARIMA(3,0,3). Or ARIMA(4,0,3).

## Series: uschange[, "Consumption"]   
## ARIMA(3,0,0) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 mean  
## 0.2274 0.1604 0.2027 0.7449  
## s.e. 0.0713 0.0723 0.0712 0.1029  
##   
## sigma^2 estimated as 0.3494: log likelihood=-165.17  
## AIC=340.34 AICc=340.67 BIC=356.5

## Series: uschange[, "Consumption"]   
## ARIMA(0,0,3) with non-zero mean   
##   
## Coefficients:  
## ma1 ma2 ma3 mean  
## 0.2403 0.2187 0.2665 0.7473  
## s.e. 0.0717 0.0719 0.0635 0.0739  
##   
## sigma^2 estimated as 0.354: log likelihood=-166.38  
## AIC=342.76 AICc=343.09 BIC=358.91

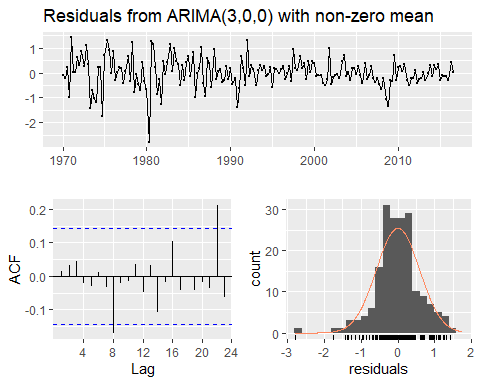
## Series: uschange[, "Consumption"]   
## ARIMA(4,0,0) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 ar4 mean  
## 0.2416 0.1714 0.2188 -0.0685 0.7461  
## s.e. 0.0727 0.0731 0.0730 0.0730 0.0965  
##   
## sigma^2 estimated as 0.3496: log likelihood=-164.73  
## AIC=341.46 AICc=341.93 BIC=360.85

## Series: uschange[, "Consumption"]   
## ARIMA(3,0,3) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 ma1 ma2 ma3 mean  
## 0.5145 0.4530 -0.3675 -0.2769 -0.3564 0.4576 0.7462  
## s.e. 0.3272 0.2457 0.2533 0.3111 0.2241 0.1678 0.0871  
##   
## sigma^2 estimated as 0.3514: log likelihood=-164.2  
## AIC=344.4 AICc=345.21 BIC=370.25

## Series: uschange[, "Consumption"]   
## ARIMA(4,0,3) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 ar4 ma1 ma2 ma3 mean  
## -0.1351 -0.4435 0.6754 0.1160 0.3973 0.7872 -0.3035 0.7461  
## s.e. 0.2039 0.0726 0.1328 0.1153 0.1968 0.1382 0.1936 0.0979  
##   
## sigma^2 estimated as 0.3366: log likelihood=-161.79  
## AIC=341.57 AICc=342.59 BIC=370.65

AICc of the ARIMA models:  
ARIMA(3,0,0) = 340.67  
ARIMA(0,0,3) = 343.09  
ARIMA(3,0,3) = 345.21  
ARIMA(4,0,0) = 341.93  
ARIMA(4,0,3) = 342.59

So out of the five models, the best fit is ARIMA(3,0,0). But what about the residuals?



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(3,0,0) with non-zero mean  
## Q\* = 6.7407, df = 4, p-value = 0.1502  
##   
## Model df: 4. Total lags used: 8

We can include a constant term that acts as a polynomial trend of order d in the forecast function.

Two options to do so in Arima():

1. *include.mean=TRUE* is the default setting. This option adds a constant term to the ARIMA model and is only valid for d = 0. Setting include.mean=FALSE allows R to set the constant term = 0.
2. *include.drift=FALSE* is the default setting. This option allows the constant term to be estimated when d=1. This option is not valid for d>1.

# Hyndman-Khandakar Algorithm for Automating ARIMA Model Selection

Step 1: Determine d by doing repeated unit root tests (KPSS).  
Step 2: Use a stepwise approach to find d and q values. Fit these initial models: \* ARIMA(0,d,0) + constant unless d = 2  
\* ARIMA(1,d,0) + constant unless d = 2  
\* ARIMA(0,d,1) + constant unless d = 2 \* ARIMA(2,d,2) + constant unless d = 2 If d <= 1, then also fit ARIMA(0,d,0) without a constant.

Step 3: Best model from above initial list becomes “current model” using AICc as performance measure.

Step 4: Vary p and q by +/-1 and including/excluding the constant for the “current model” in Step 3. Model with best AICc becomes the next “current model.”

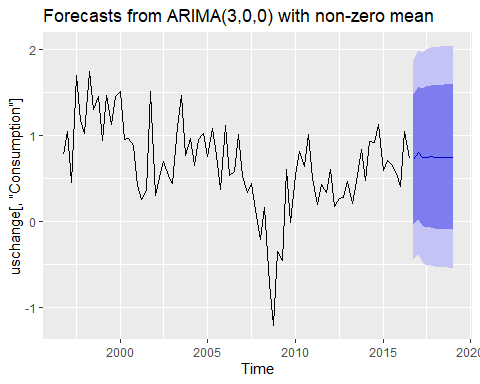
Step 5: Repeat Step 4 until AICc cannot be minimized any further.

## Series: uschange[, "Consumption"]   
## ARIMA(1,0,3) with non-zero mean   
##   
## Coefficients:  
## ar1 ma1 ma2 ma3 mean  
## 0.5885 -0.3528 0.0846 0.1739 0.7454  
## s.e. 0.1541 0.1658 0.0818 0.0843 0.0930  
##   
## sigma^2 estimated as 0.3499: log likelihood=-164.81  
## AIC=341.61 AICc=342.08 BIC=361

AICc = 342.08. ARIMA(1,0,3) does not perform as well as our ARIMA(3,0,0) model. This is because it did not account for all the possible ARIMA models. It uses a stepwise search algorithm. What if we ask auto.arima() to work harder?

## Series: uschange[, "Consumption"]   
## ARIMA(3,0,0) with non-zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 mean  
## 0.2274 0.1604 0.2027 0.7449  
## s.e. 0.0713 0.0723 0.0712 0.1029  
##   
## sigma^2 estimated as 0.3494: log likelihood=-165.17  
## AIC=340.34 AICc=340.67 BIC=356.5

Now we are in agreement with auto.arima().

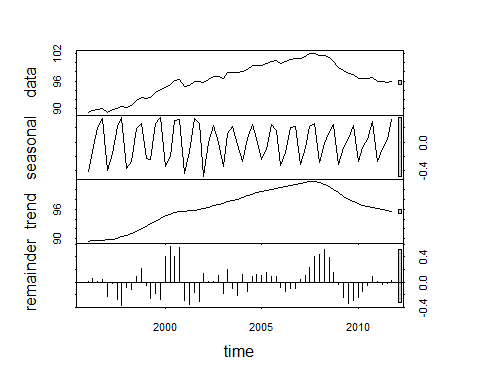
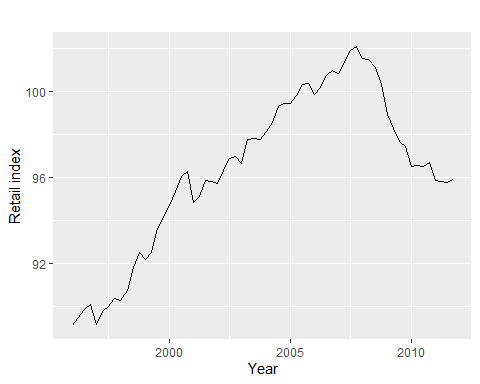


## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2016 Q4 0.7210309 -0.03647441 1.478536 -0.4374735 1.879535  
## 2017 Q1 0.7983867 0.02153619 1.575237 -0.3897036 1.986477  
## 2017 Q2 0.7501281 -0.04317044 1.543427 -0.4631173 1.963373  
## 2017 Q3 0.7498252 -0.07280963 1.572460 -0.5082862 2.007937  
## 2017 Q4 0.7576940 -0.07229199 1.587680 -0.5116600 2.027048  
## 2018 Q1 0.7496535 -0.08547881 1.584786 -0.5275711 2.026878  
## 2018 Q2 0.7490257 -0.09020785 1.588259 -0.5344712 2.032523  
## 2018 Q3 0.7491880 -0.09191440 1.590290 -0.5371671 2.035543  
## 2018 Q4 0.7474944 -0.09480031 1.589789 -0.5406842 2.035673  
## 2019 Q1 0.7470080 -0.09605465 1.590071 -0.5423451 2.036361

# Seasonal ARIMA Models

A Seasonal ARIMA model includes additional seasonal terms in the ARIMA model.  
ARIMA(p,d,q)(P,D,Q)m where the (P,D,Q)m are the seasonal components. The same autoregressive, differencing, and moving average models apply except we are now backshifting into the same seasonal period from the previous year.

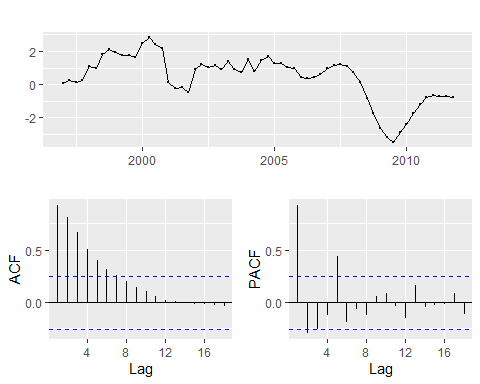
Here is a time series plot showing the quarterly retail trade index in the Euro Area (17 countries) between 1996 and 2011.

 It looks like the series has both trend/cycle and seasonality.

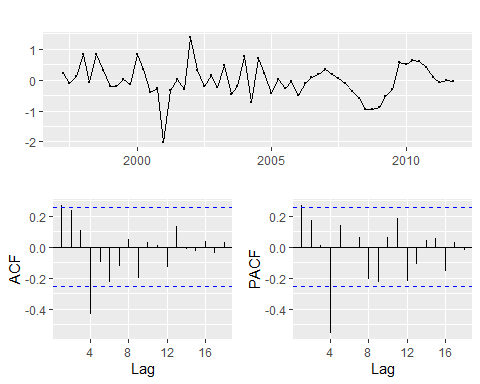
Let’s check to see if we need to seasonally difference the series.

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 3 lags.   
##   
## Value of test-statistic is: 1.1562   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

## [1] 1



## [1] 1



## [1] 0

The ACF plot shows 1st and 4th lag as significant. It is also sinuisoidal (repeating oscillatng pattern).  
The PACF plot looks the same.

Let’s tackle the ACF plot first. It looks like the first lag is the non-seasonal component and the 4th lag is the seasonal component. So our ARIMA model can be: ARIMA(0,1,1)(0,1,1)4.

Likewise, if we were to start with the PACF plot, we can say the model is: ARIMA(1,1,0)(1,1,0)4.

We can also combine AR and MA to come up with: ARIMA(1,1,1)(1,1,1)4.

Now let’s test them out.

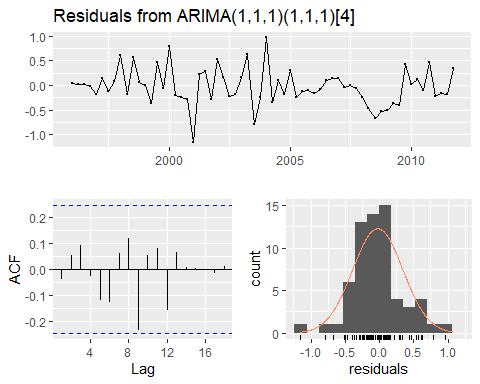
## Series: euretail   
## ARIMA(0,1,1)(0,1,1)[4]   
##   
## Coefficients:  
## ma1 sma1  
## 0.2903 -0.6913  
## s.e. 0.1118 0.1193  
##   
## sigma^2 estimated as 0.188: log likelihood=-34.64  
## AIC=75.28 AICc=75.72 BIC=81.51

## Series: euretail   
## ARIMA(1,1,0)(1,1,0)[4]   
##   
## Coefficients:  
## ar1 sar1  
## 0.4224 -0.5297  
## s.e. 0.1214 0.1095  
##   
## sigma^2 estimated as 0.1962: log likelihood=-35.29  
## AIC=76.57 AICc=77.01 BIC=82.8

## Series: euretail   
## ARIMA(1,1,1)(1,1,1)[4]   
##   
## Coefficients:  
## ar1 ma1 sar1 sma1  
## 0.8836 -0.5202 -0.0286 -0.9389  
## s.e. 0.1376 0.1715 0.1630 0.3874  
##   
## sigma^2 estimated as 0.1549: log likelihood=-30.1  
## AIC=70.2 AICc=71.33 BIC=80.59

Here are the AICc:  
ARIMA(0,1,1)(0,1,1)4 = 75.72  
ARIMA(1,1,0)(1,1,0)4 = 77.01  
ARIMA(1,1,1)(1,1,1)4 = 71.33

It looks like the last model has the best performance. Now let’s check on the residuals.



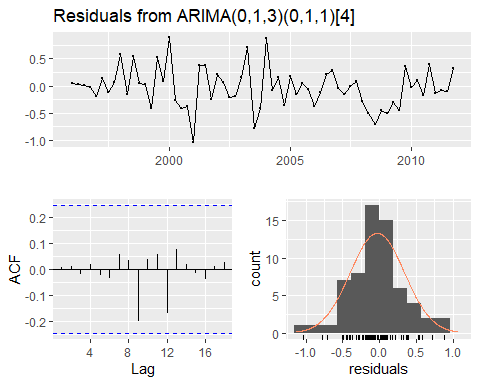
##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(1,1,1)(1,1,1)[4]  
## Q\* = 4.4884, df = 4, p-value = 0.3439  
##   
## Model df: 4. Total lags used: 8

What about auto.arima()?

## Series: euretail   
## ARIMA(0,1,3)(0,1,1)[4]   
##   
## Coefficients:  
## ma1 ma2 ma3 sma1  
## 0.2630 0.3694 0.4200 -0.6636  
## s.e. 0.1237 0.1255 0.1294 0.1545  
##   
## sigma^2 estimated as 0.156: log likelihood=-28.63  
## AIC=67.26 AICc=68.39 BIC=77.65

## Series: euretail   
## ARIMA(0,1,3)(0,1,1)[4]   
##   
## Coefficients:  
## ma1 ma2 ma3 sma1  
## 0.2630 0.3694 0.4200 -0.6636  
## s.e. 0.1237 0.1255 0.1294 0.1545  
##   
## sigma^2 estimated as 0.156: log likelihood=-28.63  
## AIC=67.26 AICc=68.39 BIC=77.65

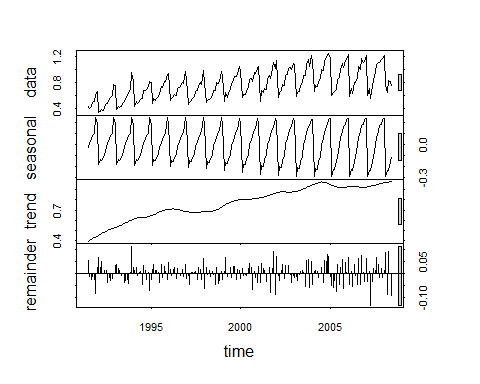
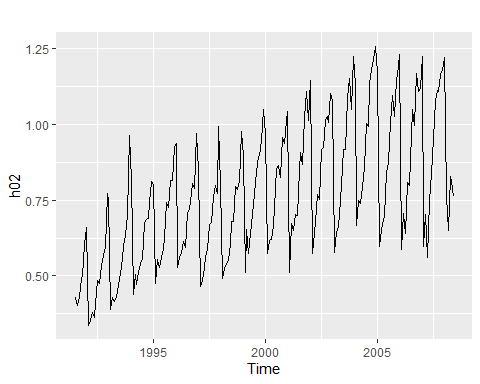
auto.arima() did even better than we did!



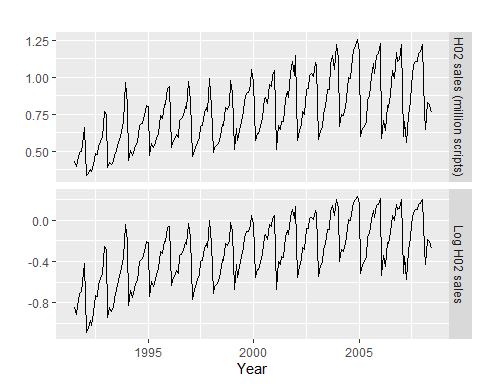
##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(0,1,3)(0,1,1)[4]  
## Q\* = 0.51128, df = 4, p-value = 0.9724  
##   
## Model df: 4. Total lags used: 8

# Another Seasonal Time Series

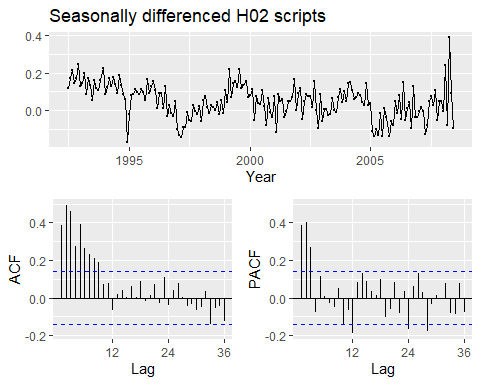
Let’s switch gears and look at another time series. This series contains the millions of monthly scripts for corticosteroid drugs in Australia from July 1991 to June 2008.



We have both trend/cycle and seasonality. The seasonal cycles do not look stable.

Let’s do a log transformation to stabilize the variance.  


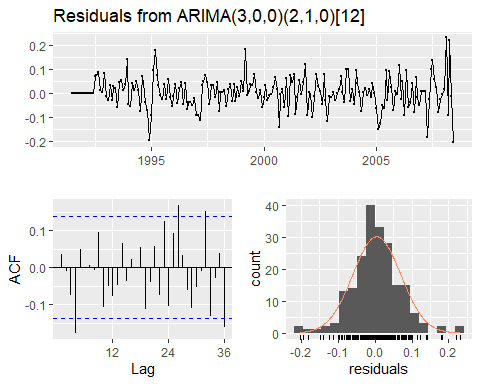
Let’s take the seasonal difference to see if it makes the mean more stable (i.e. stationary)



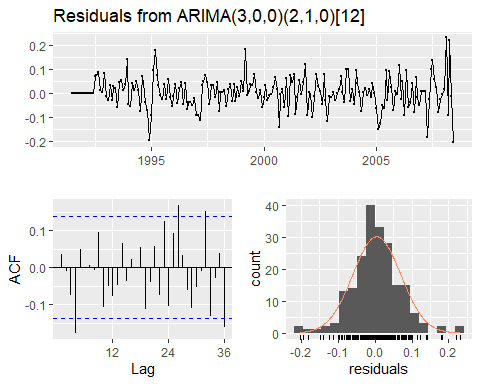
Because the ACF plot shows an exponential decay pattern and the PACF plot does not, it is likely the case that we have an autoregressive model (AR). So let’s turn our attention to the PACF plot.

This plot shows significant lags at 12 and 24, indicating there is a seasonal AR component. We may want to try AR(2) for the seasonal part. As for the non-seasonal component, we have 3 significant lags. We may want to try AR(3) for the non-seasonal component. So here is a possible ARIMA model: ARIMA(3,0,0)(2,1,0)12.

## Series: h02   
## ARIMA(3,0,0)(2,1,0)[12]   
## Box Cox transformation: lambda= 0   
##   
## Coefficients:  
## ar1 ar2 ar3 sar1 sar2  
## 0.0983 0.4060 0.4311 -0.4361 -0.290  
## s.e. 0.0695 0.0611 0.0726 0.0795 0.078  
##   
## sigma^2 estimated as 0.004622: log likelihood=243.79  
## AIC=-475.58 AICc=-475.12 BIC=-456.03



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(3,0,0)(2,1,0)[12]  
## Q\* = 28.841, df = 19, p-value = 0.06853  
##   
## Model df: 5. Total lags used: 24



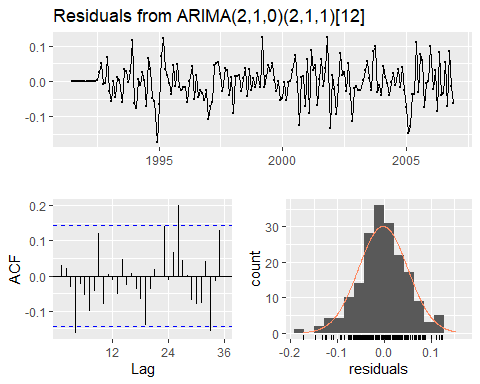
##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(3,0,0)(2,1,0)[12]  
## Q\* = 59.499, df = 31, p-value = 0.001546  
##   
## Model df: 5. Total lags used: 36

Now let’s try auto.arima()

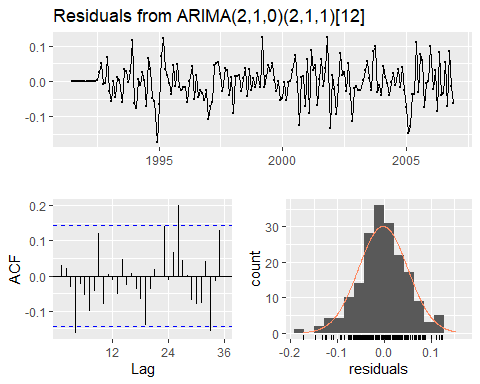
## Series: h02   
## ARIMA(2,1,1)(0,1,2)[12]   
## Box Cox transformation: lambda= 0.03147061   
##   
## Coefficients:  
## ar1 ar2 ma1 sma1 sma2  
## -1.1369 -0.5753 0.3686 -0.5329 -0.1803  
## s.e. 0.1611 0.0966 0.1890 0.0839 0.0885  
##   
## sigma^2 estimated as 0.004195: log likelihood=250.13  
## AIC=-488.26 AICc=-487.8 BIC=-468.74

# ARIMA vs. ETS()

## Series: train   
## ARIMA(2,1,0)(2,1,1)[12]   
## Box Cox transformation: lambda= 0.2995028   
##   
## Coefficients:  
## ar1 ar2 sar1 sar2 sma1  
## -0.7870 -0.3417 0.0717 -0.2458 -0.5456  
## s.e. 0.0785 0.0797 0.1413 0.0942 0.1288  
##   
## sigma^2 estimated as 0.003018: log likelihood=255.41  
## AIC=-498.81 AICc=-498.3 BIC=-479.89

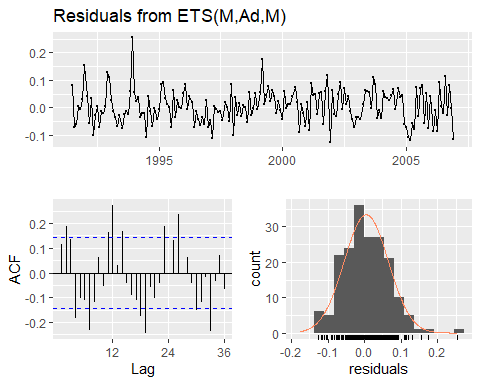


##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(2,1,0)(2,1,1)[12]  
## Q\* = 24.031, df = 19, p-value = 0.195  
##   
## Model df: 5. Total lags used: 24



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(2,1,0)(2,1,1)[12]  
## Q\* = 48.051, df = 31, p-value = 0.02602  
##   
## Model df: 5. Total lags used: 36

## ETS(M,Ad,M)   
##   
## Call:  
## ets(y = train)   
##   
## Smoothing parameters:  
## alpha = 0.2088   
## beta = 1e-04   
## gamma = 1e-04   
## phi = 0.98   
##   
## Initial states:  
## l = 0.3952   
## b = 0.0083   
## s = 0.8688 0.8233 0.767 0.7798 0.6902 1.2788  
## 1.3396 1.1763 1.1617 1.0968 1.0329 0.9848  
##   
## sigma: 0.0635  
##   
## AIC AICc BIC   
## -157.60891 -153.51310 -99.54547



##   
## Ljung-Box test  
##   
## data: Residuals from ETS(M,Ad,M)  
## Q\* = 99.053, df = 7, p-value < 2.2e-16  
##   
## Model df: 17. Total lags used: 24

## RMSE MAE MAPE MASE  
## Training set 0.04559662 0.03337967 4.295712 0.5574569  
## Test set 0.05027117 0.03837694 4.590339 0.6409137

## RMSE MAE MAPE MASE  
## Training set 0.04820710 0.03671501 4.760138 0.6131586  
## Test set 0.07120942 0.05951728 6.747825 0.9939678

ARIMA model does a better job at both fitting the train set and predicting the test set.

Now let’s use the ARIMA model to forecast the next two years.

