Exercise3

Deborah Kewon

August 10, 2019

I noticed that my previous dataset Unemployment Rate in St. Louis,USA is seasonally adjusted (no seasonal component). I will use air pollution date retrieved from Institute for Atmospheric and Climate Science instead.

source: http://data.iac.ethz.ch/CMIP6/input4MIPs/UoM/GHGConc/CMIP/mon/atmos/UoM-CMIP-1-1-0/GHGConc/ gr3-GMNHSH/v20160701/mole_fraction_of_carbon_dioxide_in_ air_input4MIPs_GHGConcentrations_CMIP_UoM-CMIP-1-1-0_gr3-GMNHSH_000001-201412.csv

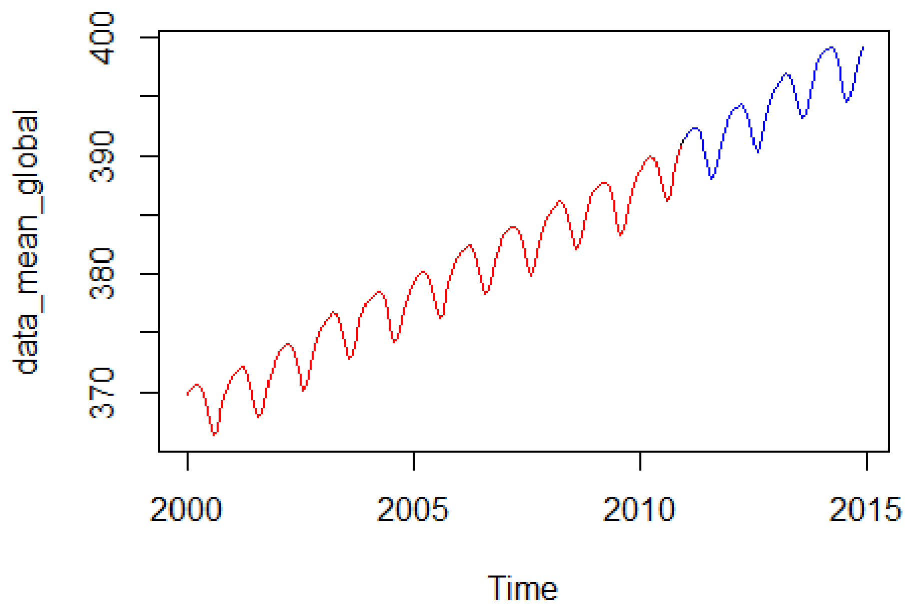
```
if (!require("fpp2")) install.packages("fpp2"); library(fpp2)
if (!require("portes")) install.packages("portes"); library(portes)
if (!require("readxl")) install.packages("readxl"); library(readxl)
if (!require("tseries")) install.packages("tseries"); library(tseries)
if (!require("lmtest")) install.packages("lmtest"); library(lmtest)
if (!require("forecast")) install.packages("forecast"); library(forecast)
if (!require("dplyr")) install.packages("dplyr"); library(dplyr)
library(readr)
options(digits=4, scipen=0)
```

1. Exploring Data

We will first look into seasonal characteristics of this data

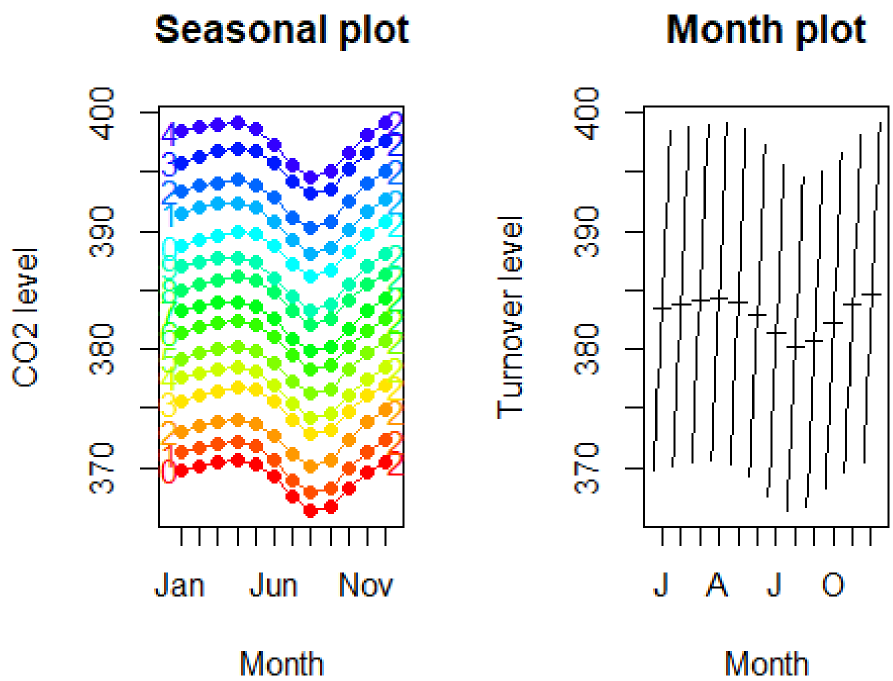
```
# Reading data
setwd('C:\\Users\\dkewon\\Desktop\\retake\\final')
df <- read_csv('Airpollution.csv')
names(df)

## [1] "datenum"          "year"             "month"
## [4] "day"              "datetime"         "data_mean_global"
## [7] "data_mean_nh"     "data_mean_sh"

#since the data includes old times, we will subset and start with 2000
subset_start_year = 2000
df_subset <- filter(df, year >= subset_start_year)
data <- ts(df_subset[,6], frequency = 1*12, start = subset_start_year)

# Split the data into training and test set
df1 <- window(data, end=c(2010,12))
df2 <- window(data, start=c(2011,1))

# Retrieve the length of the test set
```

```
h <- length(df2)

# Plot the data
par(mfrow=c(1,1))
plot(data)
lines(df1, col="red")
lines(df2, col="blue")
```



According to this graph, both seasonality and trend exist from 2000 to 2015. Month and season plots below will back up the above statement that there is moderate trend and high seasonality.

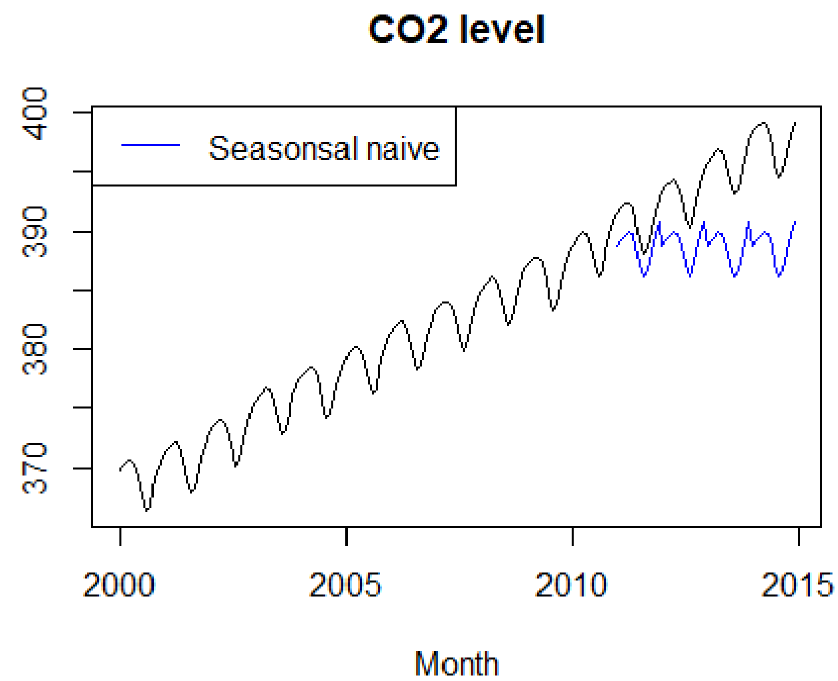
```
par(mfrow=c(1,2))
seasonplot(data, year.labels=TRUE, year.labels.left=TRUE,
           main="Seasonal plot",
           ylab="CO2 level",col=rainbow(20), pch=19)
monthplot(data, main="Month plot", ylab = "Turnover level",
          xlab="Month", type="l")
```



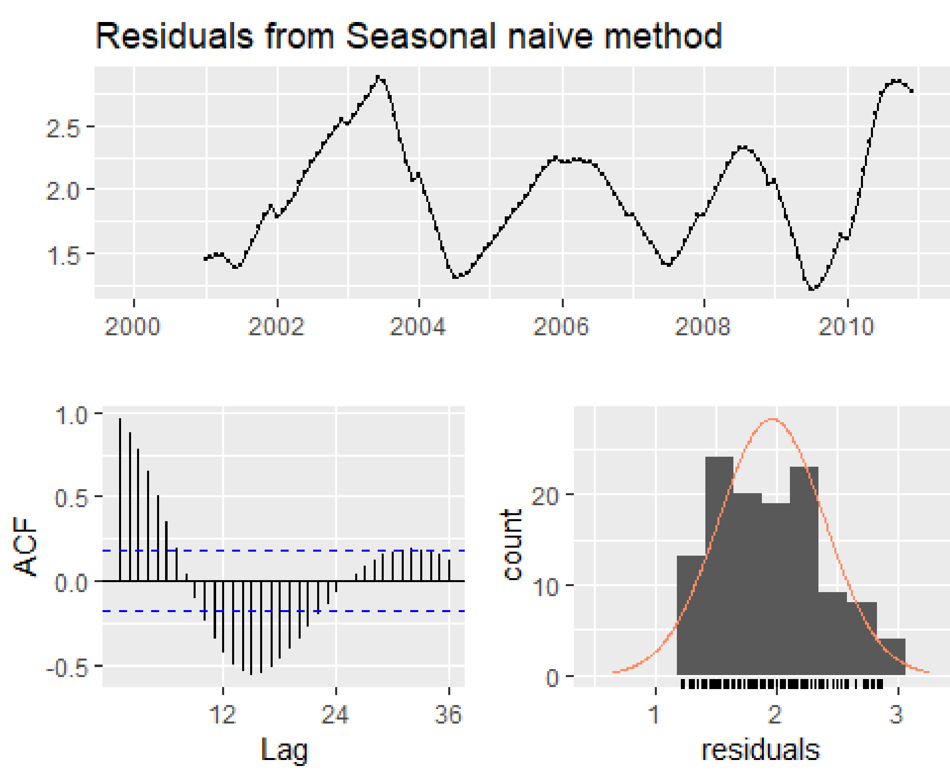
2. Seasonal Naive Method

As seasonal components exist in this data, we will only look into the seasonal naive method.

```
n <- snaive(df1, h=h) # seasonal naive
a_n <- accuracy(n,df2)[,c(2,3,5,6)]
a_train_n <- a_n[1,]
a_train_n

##   RMSE    MAE   MAPE   MASE
## 2.0074 1.9592 0.5158 1.0000

a_test_n <- a_n[2,]
a_test_n

##  RMSE   MAE  MAPE  MASE
## 6.143 5.591 1.414 2.854

par(mfrow=c(1,1))
plot(data,main="CO2 level", ylab="",xlab="Month")
lines(n$mean,col=4)
legend("topleft",lty=1,col=c(4),legend=c("Seasonsal naive"))
```




```
res <- residuals(n)
checkresiduals(n)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 740, df = 24, p-value <2e-16
##
## Model df: 0.   Total lags used: 24
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags statistic df p-value
##     1     113.1  1       0
##     5     369.1  5       0
##     9     391.1  9       0
##    13     474.9 13       0
##    17     640.2 17       0
##    21     725.2 21       0
```

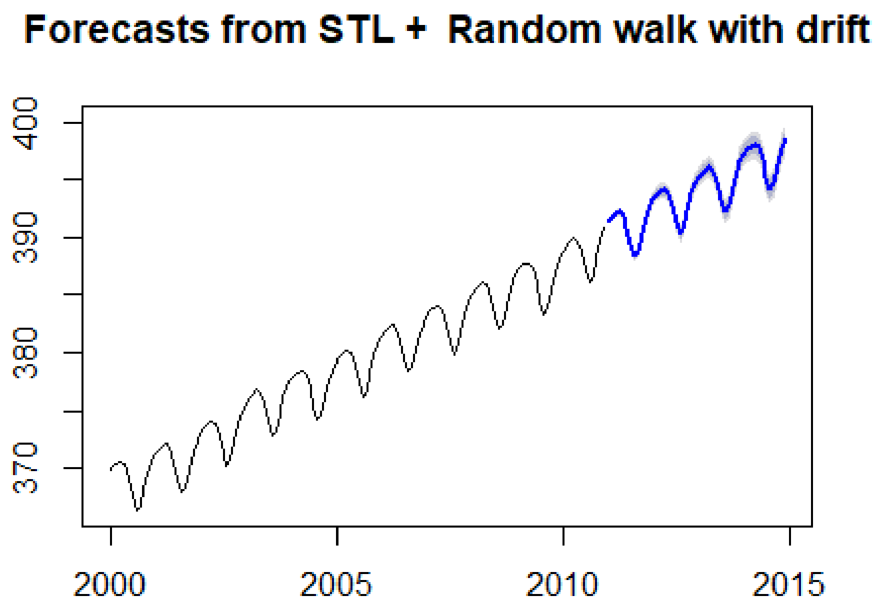
The graph retrieved from the seasonal naive method looks like it has seasonal components. However, it doesn't completely correspond with the general trend,which is constantly increasing.

The residual diagnostics show that the residuals of this naive method do not contain white noise; there is still information in the residual.

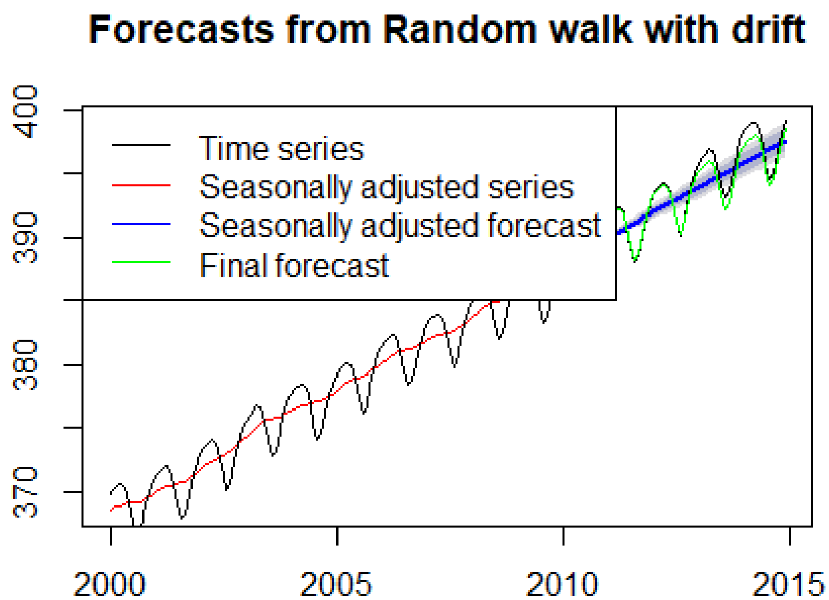
3.STL Decomposition

In this section, we will use STL to seasonally adjust the data and use a random walk with drift method to forecast.

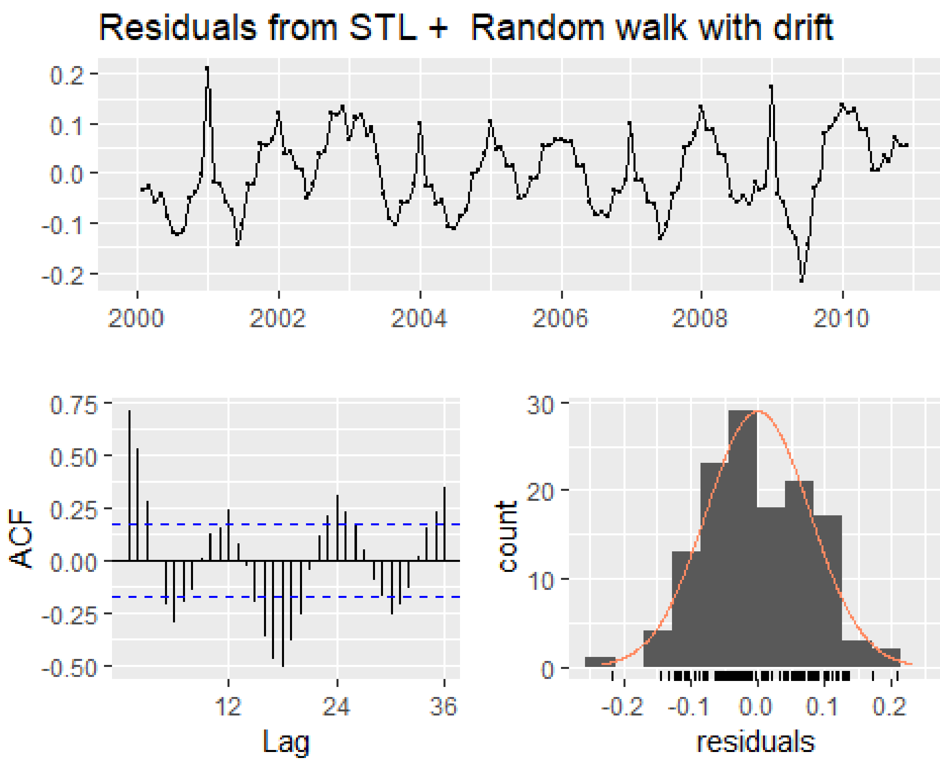
```
d <- stl(df1[,1], t.window=15, s.window=13)
dataadj <- seasadj(d)

f_d <- forecast(d, method="rwdrift", h=h)
plot(f_d)
```




```
par(mfrow=c(1,1))
plot(rwf(dataadj, drift=TRUE, h=h), col="red")
lines(data, col="black")
lines(f_d$mean, col="green")
legend("topleft", lty=1, col=c("black", "red", "blue", "green"),
       legend=c("Time series","Seasonally adjusted series",
                "Seasonally adjusted forecast", "Final forecast"))
```




```
# We check the accuracy of the forecasts
a_d <- accuracy(f_d,df2)[,c(2,3,5,6)]
a_train_d <- a_d[1,]
a_train_d

##     RMSE      MAE     MAPE     MASE
## 0.07694 0.06434 0.01698 0.03284

a_test_d <- a_d[2,]
a_test_d

##    RMSE     MAE    MAPE    MASE
## 0.6222 0.4955 0.1251 0.2529

# We also check the residuals of the STL method.
checkresiduals(f_d)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk with drift
## Q* = 320, df = 23, p-value <2e-16
##
## Model df: 1.   Total lags used: 24
```

```
res <- na.omit(f_d$residuals)
LjungBox(res, lags=seq(1,24,4), order=1)
```

```
##  lags statistic df p-value
##     1     67.77  0       0
##     5    122.31  4       0
##     9    143.36  8       0
##    13    158.06 12       0
##    17    218.96 16       0
##    21    293.08 20       0
```

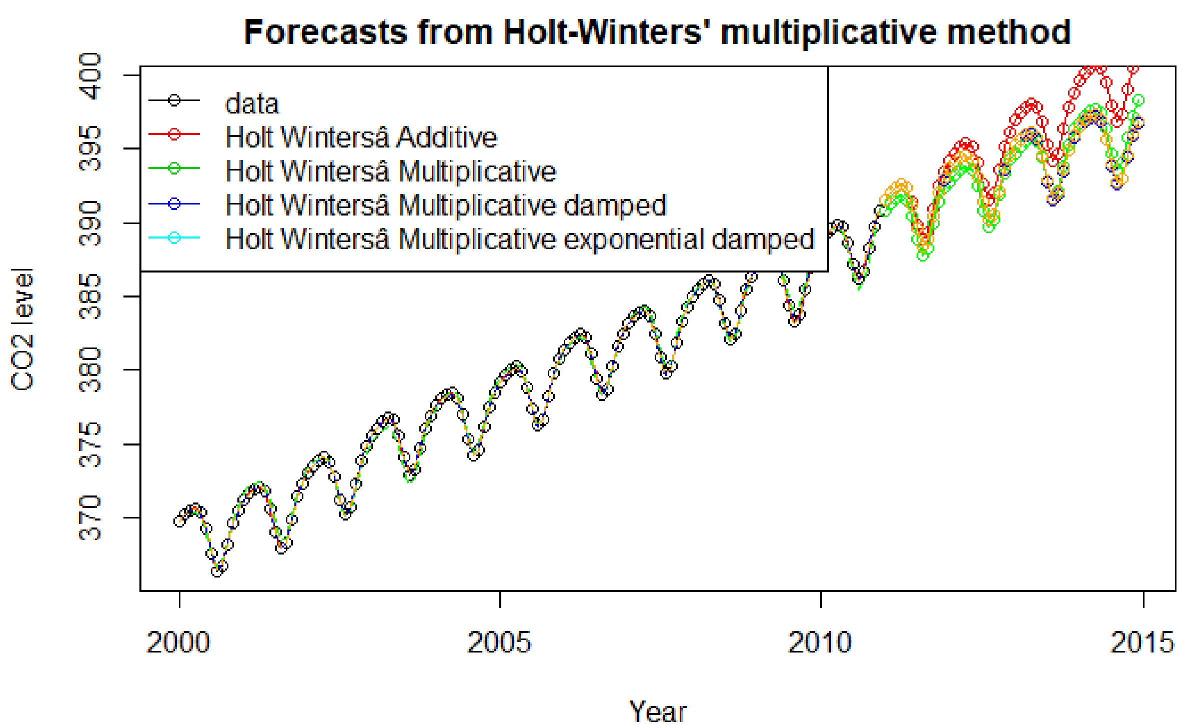
There is no white noise.There is still information we can capture.

4. Holt Winter's Method

In this section, we tested 4 different holt winter's methods. Among these methods, multiplicative hw method performs the best in the test set in terms of accuracy.

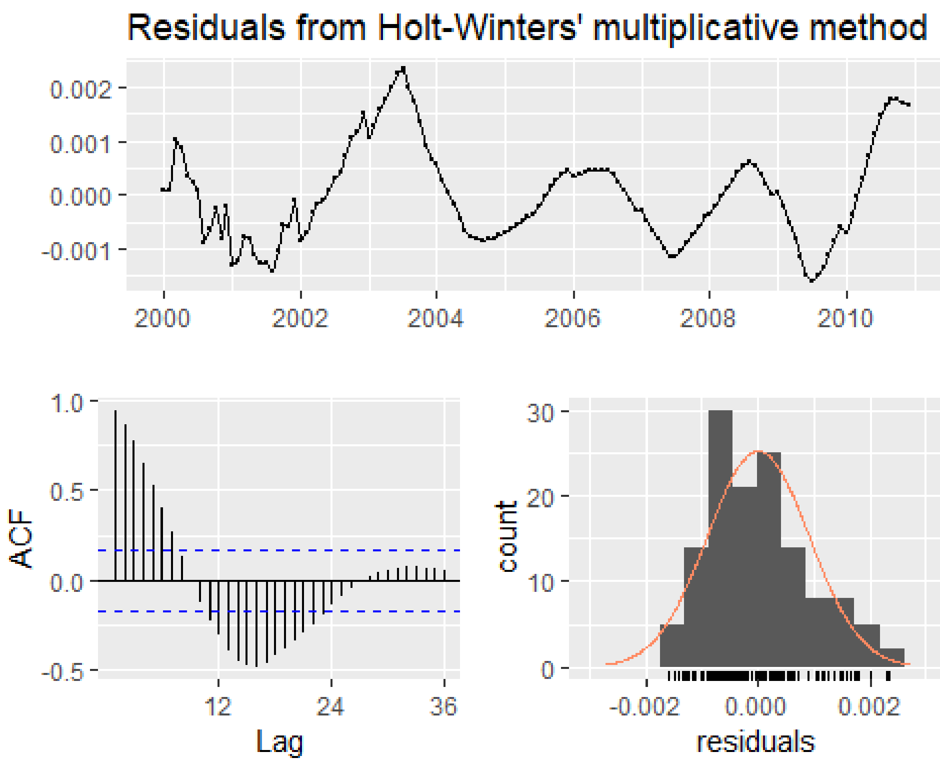
As for model fit AIC, additive hw method performs well.

```
fit1 <- hw(df1,seasonal="additive",h=h)
fit2 <- hw(df1,seasonal="multiplicative",h=h)
fit3 <- hw(df1,seasonal="multiplicative", damped=TRUE,h=h)
fit4 <- hw(df1,seasonal="multiplicative",exponential=TRUE, damped=TRUE,h=h)

plot(fit2,ylab="CO2 level",
     shadecols = "white",
     type="o", fcol="white", xlab="Year")
lines(fitted(fit1), col="red", lty=2)
lines(fitted(fit2), col="green", lty=2)
lines(fitted(fit3), col="blue", lty=2)
lines(fitted(fit4), col="orange", lty=2)
lines(fit1$mean, type="o",  col="red")
lines(fit2$mean, type="o",  col="green")
lines(fit3$mean, type="o",  col="blue")
lines(fit4$mean, type="o",  col="orange")
legend("topleft",lty=1, pch=1, col=1:5,
       c("data",
         "Holt Winters Additive",
         "Holt Winters Multiplicative",
         "Holt Winters Multiplicative damped",
         "Holt Winters Multiplicative exponential damped"))
```




```
# check acc with its own train set
a_fc1 <- accuracy(fit1)[,c(2,3,5,6)]
```

```r
a_fc2 <- accuracy(fit2)[,c(2,3,5,6)]
a_fc3 <- accuracy(fit3)[,c(2,3,5,6)]
a_fc4 <- accuracy(fit4)[,c(2,3,5,6)]

acc <- rbind(a_fc1, a_fc2, a_fc3, a_fc4)
rownames(acc) <- c("a_fc1", "a_fc2", "a_fc3", "a_fc4")
acc

##          RMSE     MAE    MAPE    MASE
## a_fc1 0.05743 0.04386 0.01157 0.02239
## a_fc2 0.34150 0.27667 0.07305 0.14122
## a_fc3 0.07407 0.05851 0.01545 0.02987
## a_fc4 0.07254 0.05665 0.01495 0.02891

# check acc with test set
a_fc1 <- accuracy(fit1, df2)[,c(2,3,5,6)]
a_fc2 <- accuracy(fit2, df2)[,c(2,3,5,6)]
a_fc3 <- accuracy(fit3, df2)[,c(2,3,5,6)]
a_fc4 <- accuracy(fit4, df2)[,c(2,3,5,6)]

acc <- rbind(a_fc1, a_fc2, a_fc3, a_fc4)
# rownames(acc) <- c("a_fc1", "a_fc2", "a_fc3", "a_fc4")
acc

##                 RMSE     MAE    MAPE    MASE
## Training set 0.05743 0.04386 0.01157 0.02239
## Test set     1.33646 1.19562 0.30257 0.61026
## Training set 0.34150 0.27667 0.07305 0.14122
## Test set     0.93364 0.82839 0.20946 0.42282
## Training set 0.07407 0.05851 0.01545 0.02987
## Test set     1.21552 0.94797 0.23933 0.48385
## Training set 0.07254 0.05665 0.01495 0.02891
## Test set     1.15869 0.90742 0.22911 0.46315

# a_fc2 performs best in the test set in terms of RMSE, MAE, MAPE and MASE

fit <- rbind(fit1$model$aic, fit2$model$aic, fit3$model$aic, fit4$model$aic)
colnames(fit) <- c("AIC")
rownames(fit) <- c("a_fc1", "a_fc2", "a_fc3", "a_fc4")
fit

##           AIC
## a_fc1 -75.747
## a_fc2 395.042
## a_fc3  -6.389
## a_fc4 -12.136

# a_fc1 shows the lowest AIC

checkresiduals(fit2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 710, df = 8, p-value <2e-16
##
## Model df: 16.   Total lags used: 24
```

```
res <- na.omit(f_d$residuals)
LjungBox(res, lags=seq(1,24,4), order=1)
```

```
##  lags statistic df p-value
##     1     67.77  0       0
##     5    122.31  4       0
##     9    143.36  8       0
##    13    158.06 12       0
##    17    218.96 16       0
##    21    293.08 20       0
```

These residuals still show remaining autocorrelation. There is no white noise.

5.ETS

Considering that the data contains seasonal components, we will test multiple ETS methods such as additive and multiplicative, with and without damping.

```
#Models without damping
e1 <- ets(df1, model="AAA", damped=FALSE)
e2 <- ets(df1, model="MAA", damped=FALSE)
e3 <- ets(df1, model="MAM", damped=FALSE)
e4 <- ets(df1, model="MMM", damped=FALSE)
#Models with damping
e5 <- ets(df1, model="AAA", damped=TRUE)
e6 <- ets(df1, model="MAA", damped=TRUE)
e7 <- ets(df1, model="MAM", damped=TRUE)
e8 <- ets(df1, model="MMM", damped=TRUE)
```

We will consider AICc for model fit and RMSE,MAE, MAPE and MASE for accuracy.

```
m <- c("AAA", "MAA", "MAM", "MMM")
result <- matrix(data=NA, nrow=4, ncol=9)
for (i in 1:4){
  model <- ets(df1, model=m[i], damped=FALSE)
  f <- forecast(model, h=length(df2))
  a <- accuracy(f, df2)
  result[i,1] <- model$aicc
  result[i,2] <- a[1,2]
  result[i,3] <- a[1,3]
  result[i,4] <- a[1,5]
  result[i,5] <- a[1,6]
  result[i,6] <- a[2,2]
  result[i,7] <- a[2,3]
  result[i,8] <- a[2,5]
  result[i,9] <- a[2,6]
}
rownames(result) <- m
result[,1] # Compare AICc values

##    AAA    MAA    MAM    MMM
## -70.38 -68.54 400.87 405.72

a_train_e1 <- result[,2:5]
colnames(a_train_e1) <- c("RMSE", "MAE", "MAPE", "MASE")
a_train_e1

##        RMSE     MAE    MAPE    MASE
## AAA 0.05743 0.04386 0.01157 0.02239
## MAA 0.05795 0.04311 0.01137 0.02201
## MAM 0.34158 0.27732 0.07329 0.14155
## MMM 0.34763 0.28170 0.07448 0.14378
```

```
a_test_e1 <- result[,6:9]
colnames(a_test_e1) <- c("RMSE", "MAE", "MAPE", "MASE")
a_test_e1
```

```
##       RMSE    MAE   MAPE   MASE
## AAA 1.3365 1.1956 0.3026 0.6103
## MAA 1.3397 1.1990 0.3034 0.6120
## MAM 0.9530 0.8512 0.2152 0.4345
## MMM 0.7792 0.6831 0.1727 0.3487
```

The non-damped MAM model shows the best AICc. However, in terms of accuracy, the non-damped MMM model has the lowest error values (for the test set).Now we will apply the same procedure for the damped one.

```
m <- c("AAA", "MAA", "MAM", "MMM")
result <- matrix(data=NA, nrow=4, ncol=9)
for (i in 1:4){
  model <- ets(df1, model=m[i], damped=TRUE)
  f <- forecast(model, h=length(df2))
  a <- accuracy(f, df2)
  result[i,1] <- model$aicc
  result[i,2] <- a[1,2]
  result[i,3] <- a[1,3]
  result[i,4] <- a[1,5]
  result[i,5] <- a[1,6]
  result[i,6] <- a[2,2]
  result[i,7] <- a[2,3]
  result[i,8] <- a[2,5]
  result[i,9] <- a[2,6]
}
rownames(result) <- c("AAdA", "MAdA", "MAdM", "MMdM")
result[,1] # Compare AICc values
```

```
##     AAdA     MAdA     MAdM     MMdM
## -40.5693 -16.9029   0.4752  -2.7071
```

```
a_train_e2 <- result[,2:5]
colnames(a_train_e2) <- c("RMSE", "MAE", "MAPE", "MASE")
a_train_e2
```

```
##         RMSE     MAE    MAPE    MASE
## AAdA 0.06365 0.05163 0.01363 0.02635
## MAdA 0.06966 0.05406 0.01428 0.02759
## MAdM 0.07446 0.06183 0.01630 0.03156
## MMdM 0.07355 0.05712 0.01507 0.02916
```

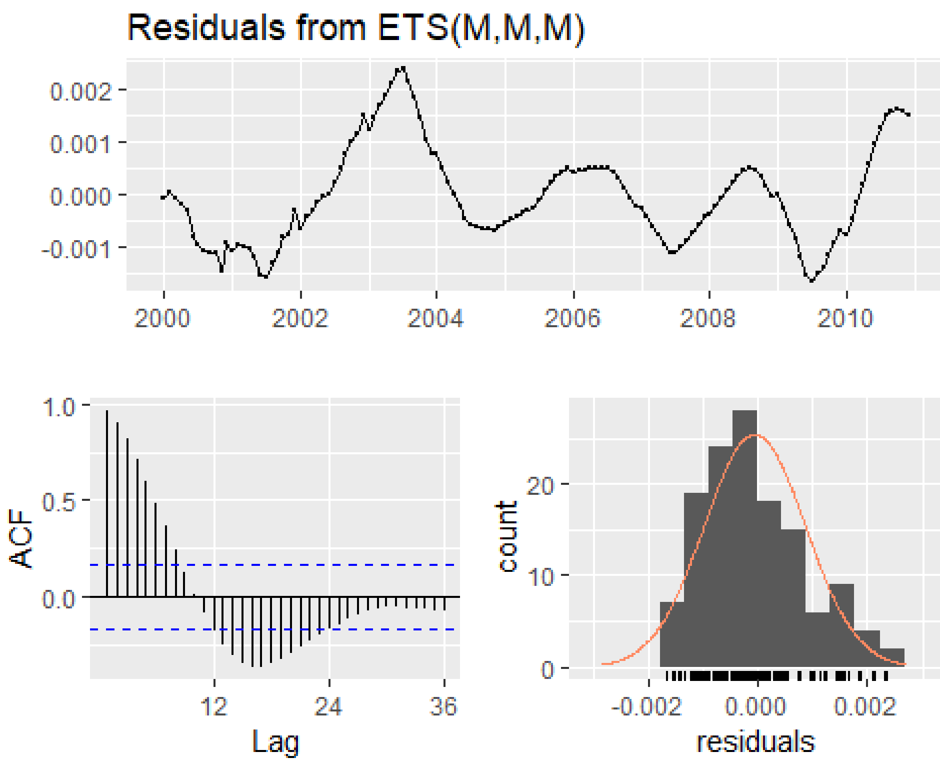
```
a_test_e2 <- result[,6:9]
colnames(a_test_e2) <- c("RMSE", "MAE", "MAPE", "MASE")
a_test_e2
```

```
##         RMSE    MAE   MAPE   MASE
## AAdA 1.1122 0.8704 0.2197 0.4443
## MAdA 0.8723 0.7306 0.1847 0.3729
## MAdM 1.1883 0.9149 0.2309 0.4670
## MMdM 1.0492 0.8430 0.2130 0.4303
```

```
# We select the non-damped MMM model considering low error terms on the test set.
summary(e4)
```

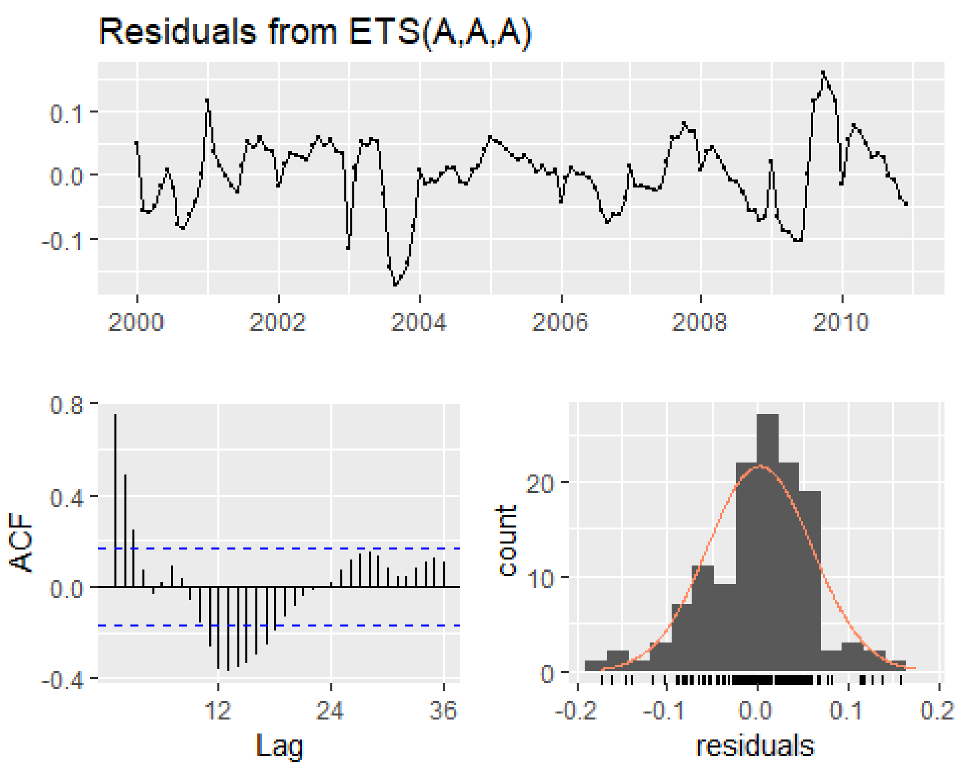
```
## ETS(M,M,M)
##
## Call:
##  ets(y = df1, model = "MMM", damped = FALSE)
##
##   Smoothing parameters:
##     alpha = 0.0187
##     beta  = 1e-04
##     gamma = 0.472
##
##   Initial states:
##     l = 368.3665
##     b = 1.0004
##     s = 1.002 1 0.9965 0.9929 0.9923 0.9959
##            1.001 1.004 1.005 1.004 1.004 1.004
##
##   sigma:  0.001
##
##   AIC  AICc   BIC
## 400.3 405.7 449.4
##
## Training set error measures:
##                     ME   RMSE    MAE       MPE    MAPE   MASE  ACF1
## Training set -0.0216 0.3476 0.2817 -0.005893 0.07448 0.1438 0.965
```

```
# We check the properties of the residuals for this model.
checkresiduals(e4)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,M,M)
## Q* = 680, df = 8, p-value <2e-16
##
## Model df: 16.   Total lags used: 24

res <- na.omit(e4$residuals)
LjungBox(res, lags = seq(length(e4$par),24,4), order=length(e4$par))

##  lags statistic df p-value
##    16     581.4  0       0
##    20     651.6  4       0
##    24     684.0  8       0

# we reject the null hypothesis of white noise.
# We will compare the results with those of the automated ETS procedure.

auto_ets <- ets(df1)
auto_ets$method

## [1] "ETS(A,A,A)"

f <- forecast(auto_ets, h=length(df2))
accuracy(f, df2)[,c(2,6)]
```

```
##                     RMSE    MASE
## Training set 0.05743 0.02239
## Test set     1.33646 0.61026

checkresiduals(auto_ets)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,A,A)
## Q* = 240, df = 8, p-value <2e-16
##
## Model df: 16.   Total lags used: 24
```

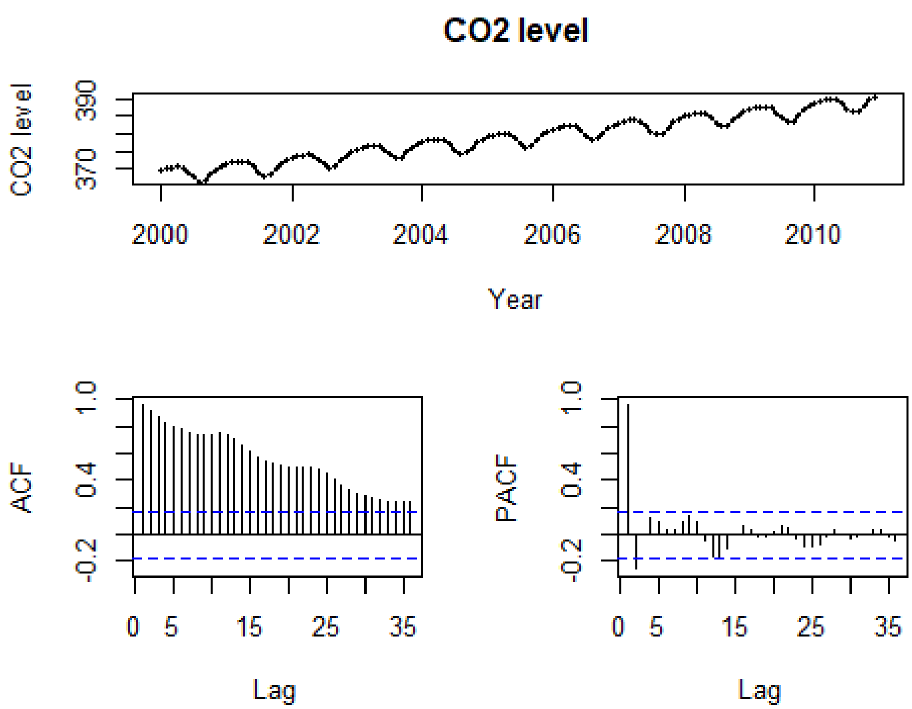
The ETS(A,A,A) model from auto ets has the best fit (best AICc). However, this is not the model with the best performance in terms of forecast accuracy.

Even though there is no white noise, we choose ETS(M,M,M) model (e4) as a best model to forecast based on accuracy metrics.

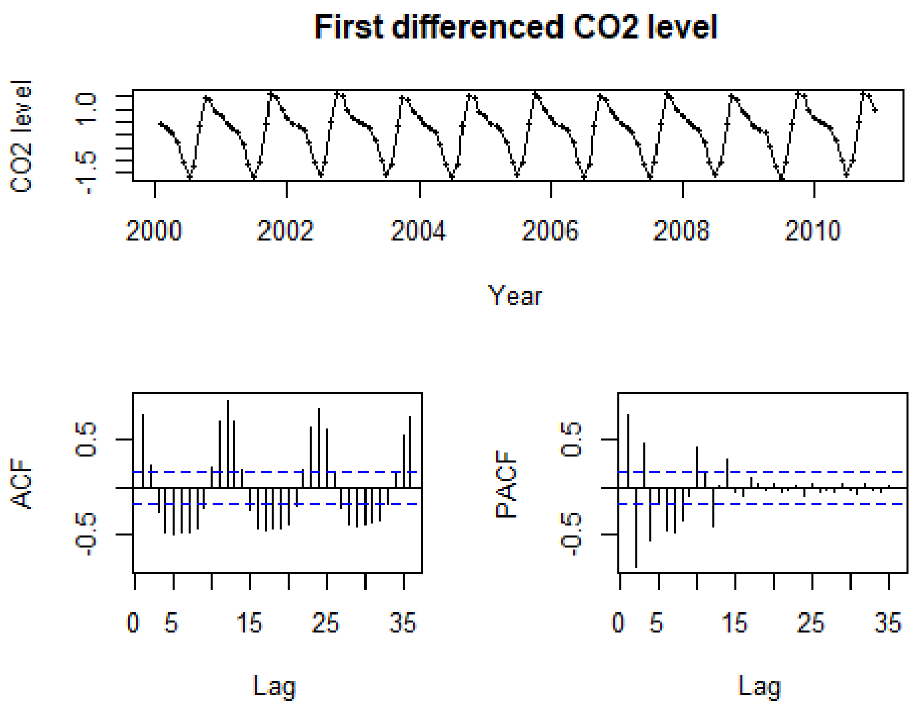
```
e4 <- ets(df1, model="MMM",damped = FALSE)
f_e4 <- forecast(e4, h=length(df2))
a_e4 <- accuracy(f_e4,df2)[,c(2,3,5,6)]
```

6. ARIMA

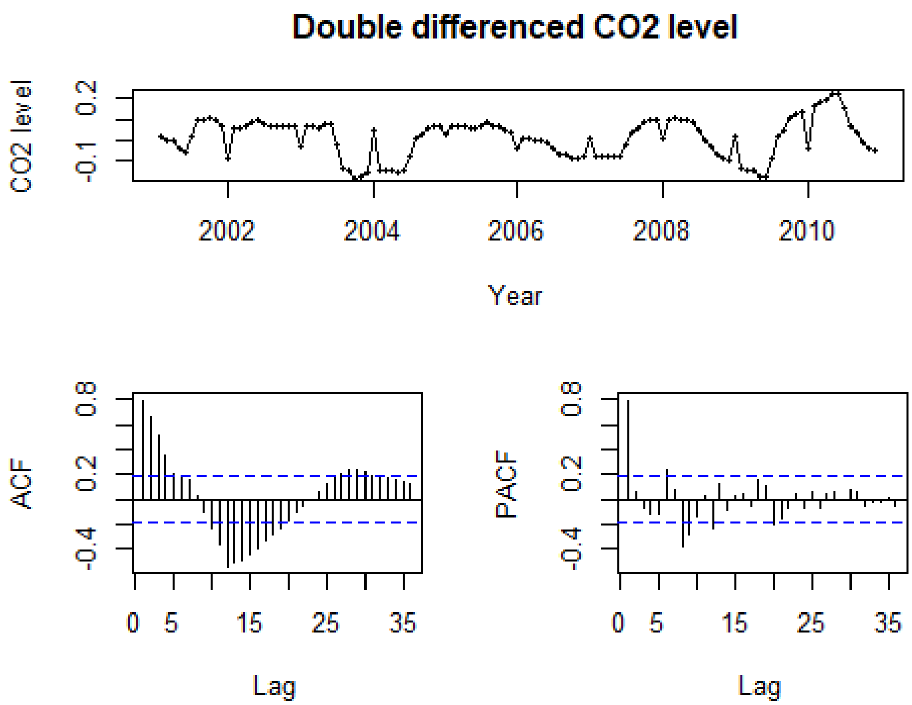
```r
tsdisplay(df1, main="CO2 level", ylab="CO2 level", xlab="Year")
```




```r
#The ACF shows that it is nonstationary. We start by differencing the data.

ndiffs(df1)

## [1] 1

tsdisplay((diff(df1)), main="First differenced CO2 level",
          ylab="CO2 level", xlab="Year")
```




```r
# The nsdiffs function also proposes to take seasonal differences.
nsdiffs(diff(df1))

## [1] 1

# As the data is seasonal, we take seasonal differences.
tsdisplay(diff(diff(df1,12)), main="Double differenced CO2 level",
          ylab="CO2 level", xlab="Year")
```

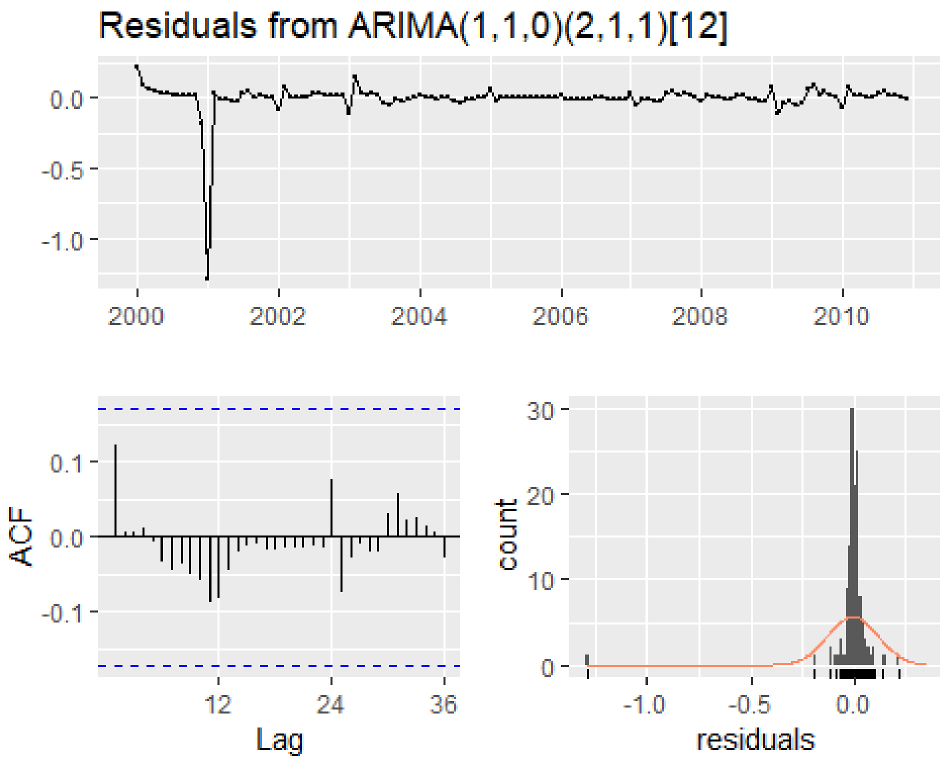



```
# 6.2 Model estimation
getinfo <- function(x,h,...){
  train.end <- time(x)[length(x)-h]
  test.start <- time(x)[length(x)-h+1]
  train <- window(x,end=train.end)
  test <- window(x,start=test.start)
  fit <- Arima(train,...)
  fc <- forecast(fit,h=h)
  a <- accuracy(fc,test)
  result <- matrix(NA, nrow=1, ncol=5)
  result[1,1] <- fit$aicc
  result[1,2] <- a[1,6]
  result[1,3] <- a[2,6]
  result[1,4] <- a[1,2]
  result[1,5] <- a[2,2]
  return(result)
}

mat <- matrix(NA,nrow=54, ncol=5)
modelnames <- vector(mode="character", length=54)
line <- 0
for (i in 0:2){
  for (j in 0:2){
    for (k in 0:1){
      for (l in 0:2){
        line <- line+1
```

```
        mat[line,] <- getinfo(data,h=h,order=c(i,1,j),seasonal=c(k,1,l))
        modelnames[line] <- paste0("ARIMA(",i,",1,",j,")(",k,",1,",l,")[12]")
      }
    }
  }
}


colnames(mat) <- c("AICc", "MASE_train", "MASE_test", "RMSE_train", "RMSE_tes
t")
rownames(mat) <- modelnames

#save as dataframe
mat_df = as.data.frame(mat)
mat_df['modelnames']=modelnames

# we will mainly focus on AICc and MASE/ RMSE on test set

# best AICc
mat_df[mat_df['AICc']==min(mat_df['AICc'])]

## [1] "-428.6"                        "0.01883"
## [3] "0.2086"                        "0.1200"
## [5] "0.5334"                        "ARIMA(1,1,0)(0,1,2)[12]"

# best MASE_train
mat_df[mat_df['MASE_train']==min(mat_df['MASE_train'])]

## [1] "-422.9"                        "0.01842"
## [3] "0.3292"                        "0.1196"
## [5] "0.7996"                        "ARIMA(2,1,1)(1,1,1)[12]"

# best RMSE_test
mat_df[mat_df['RMSE_test']==min(mat_df['RMSE_test'])]

## [1] "-278.6"                        "0.03678"
## [3] "0.1321"                        "0.1340"
## [5] "0.3381"                        "ARIMA(0,1,0)(1,1,0)[12]"

# We continue with the auto.arima procedure
m0 <- auto.arima(df1, stepwise = FALSE, approximation = FALSE, d=1, D=1)
m0

## Series: df1
## ARIMA(1,1,0)(2,1,1)[12]
##
## Coefficients:
##          ar1     sar1     sar2     sma1
##        0.779   -0.582   -0.383   -0.651
## s.e.   0.053    0.161    0.135    0.170
##
```

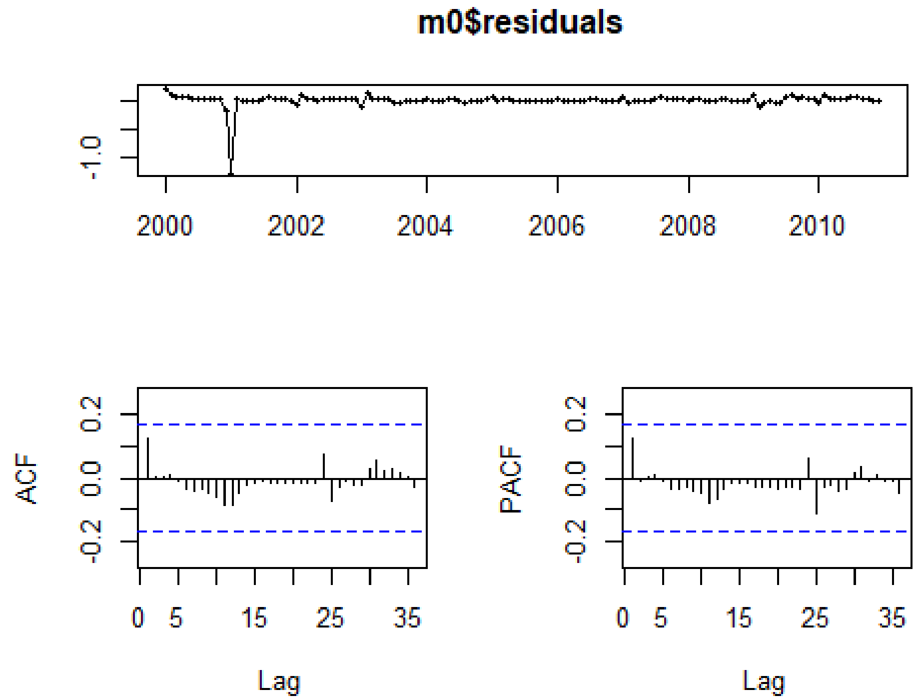
```
## sigma^2 estimated as 0.0165:  log likelihood=220.5
## AIC=-431   AICc=-430.4   BIC=-417.1
```

```
checkresiduals(m0)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(2,1,1)[12]
## Q* = 7.3, df = 20, p-value = 1
##
## Model df: 4.   Total lags used: 24
```

```
tsdisplay(m0$residuals)
```




```
LjungBox(m0$residuals, lags=seq(length(m0$coef),24,4), order=length(m0$coef))

##  lags statistic df p-value
##     4     2.068  0  0.0000
##     8     2.701  4  0.6090
##    12     5.684  8  0.6826
##    16     6.086 12  0.9117
##    20     6.250 16  0.9852
##    24     7.318 20  0.9955

f0 <- forecast(m0, h=h)
accuracy(f0,df2)[,c(2,3,5,6)]

##                 RMSE     MAE     MAPE    MASE
## Training set 0.1198 0.03629 0.009649 0.01852
## Test set     0.5185 0.40784 0.103077 0.20817
```

Based on the above results, three models are selected; 1) m0: ARIMA(1,1,0)(2,1,1) acceptable fit with white noise 2) m1: ARIMA(1,1,0)(0,1,2) best AICc. 3) m2: ARIMA(0,1,0)(1,1,0) best MASE and RMSE on the test set.

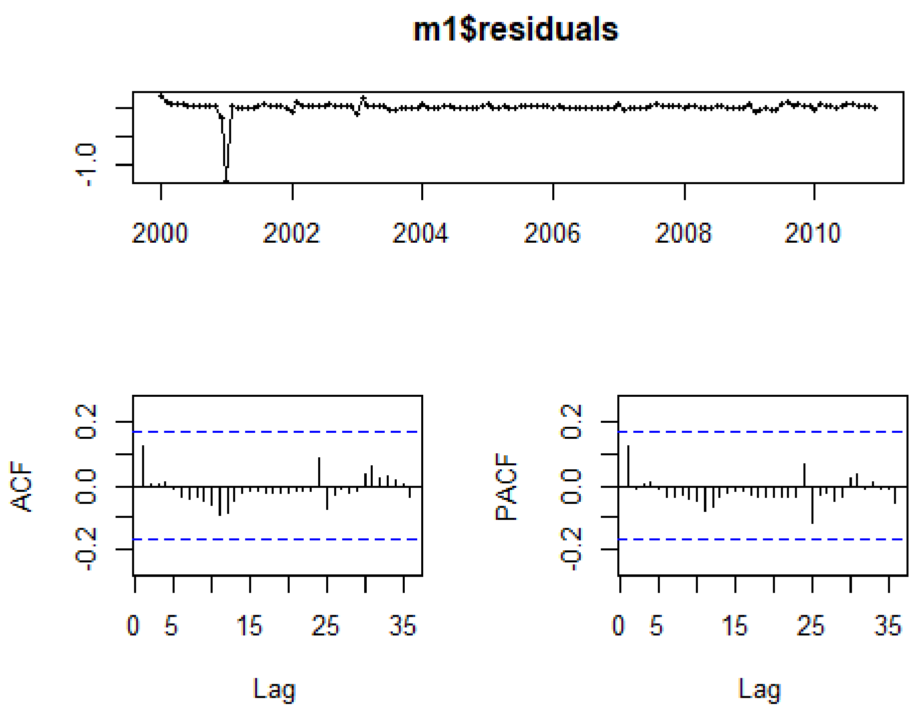
```
m1 <- Arima(df1, order=c(1,1,0), seasonal=c(0,1,2))
coeftest(m1)

##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
```

```
## ar1     0.7748     0.0543   14.28  < 2e-16 ***
## sma1  -1.2390     0.0985  -12.58  < 2e-16 ***
## sma2   0.3965     0.1091    3.63  0.00028 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
LjungBox(m1$residuals, lags=seq(length(m1$coef),24,4), order=length(m1$coef))
```

```
##  lags statistic df p-value
##     3     2.087  0  0.0000
##     7     2.530  4  0.6393
##    11     4.700  8  0.7891
##    15     6.184 12  0.9065
##    19     6.411 16  0.9830
##    23     6.582 20  0.9978
```

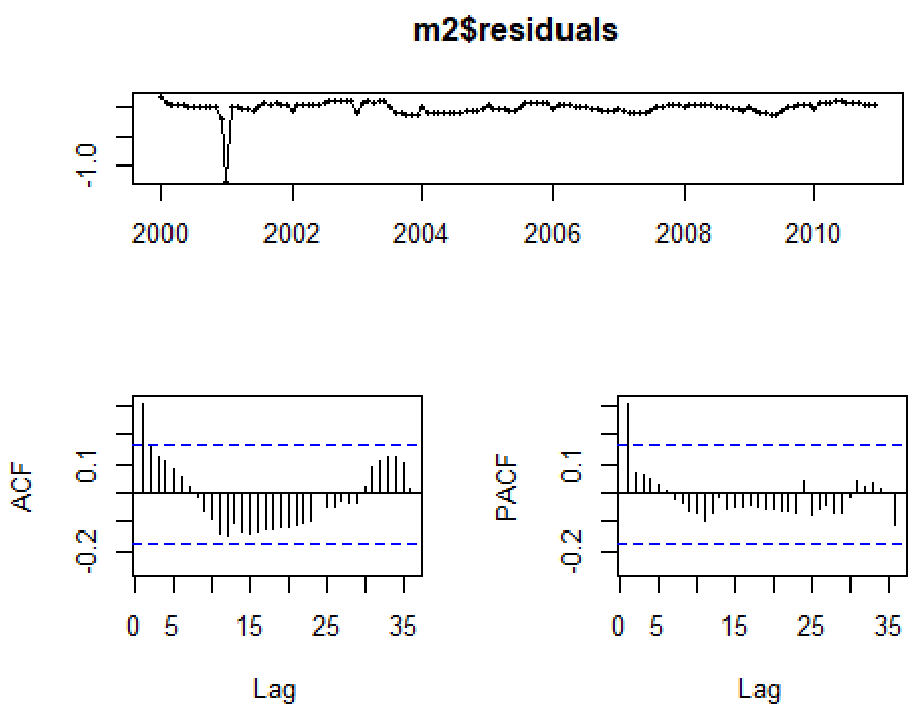
```
# the requirements of white noise residuals are fulfilled in m1.

tsdisplay(m1$residuals)
```




```
f1 <- forecast(m1, h=h)

m2 <- Arima(df1, order=c(0,1,0), seasonal=c(1,1,0))
coeftest(m2)
```

```
##
## z test of coefficients:
```

```
##
##      Estimate Std. Error z value Pr(>|z|)
## sar1  -0.6456     0.0736   -8.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

LjungBox(m2$residuals, lags=seq(length(m2$coef),24,4), order=length(m2$coef))

##  lags statistic df   p-value
##     1     12.90  0 0.0000000
##     5     21.32  4 0.0002732
##     9     22.41  8 0.0042163
##    13     31.46 12 0.0016731
##    17     42.06 16 0.0003872
##    21     50.72 20 0.0001743

# We observe that the requirements of white noise residuals are not fulfilled
in m2.

tsdisplay(m2$residuals)
```




```
f2 <- forecast(m2, h=h)

a_m0 <- accuracy(f0,df2)[,c(2,3,5,6)]
a_m1 <- accuracy(f1,df2)[,c(2,3,5,6)]
a_m2 <- accuracy(f2,df2)[,c(2,3,5,6)]
```

```
a_train_a <- rbind(a_m0[1,], a_m1[1,], a_m2[1,])
rownames(a_train_a) <- c("a_m0", "a_m1", "a_m2")
a_train_a
```

```
##          RMSE     MAE     MAPE    MASE
## a_m0 0.1198 0.03629 0.009649 0.01852
## a_m1 0.1200 0.03690 0.009816 0.01883
## a_m2 0.1340 0.07205 0.019089 0.03678
```

```
a_test_a <- rbind(a_m0[2,], a_m1[2,], a_m2[2,])
rownames(a_test_a) <- c("a_m0", "a_m1", "a_m2")
a_test_a
```

```
##          RMSE    MAE    MAPE   MASE
## a_m0 0.5185 0.4078 0.10308 0.2082
## a_m1 0.5334 0.4087 0.10322 0.2086
## a_m2 0.3381 0.2588 0.06555 0.1321
```

Even though model m2 doesn't have white noise, it has the lowest error terms (RMSE, MAE, MAPE AND MASE) for the test set. For this reason, we select m2:Arima (0,1,0)(1,1,0) as a final model.

7. Final Model

In this section, we will compare the performance of seasonal naive, the STL decomposition, the Holt-Winters method, the ets procedure and ARIMA.

```
final_train <- rbind(a_train_n, a_train_d, a_fc2[1,], a_e4[1,], a_m0[1,])
rownames(final_train) <- c("snaive", "decompose","Holt-Winters", "ETS(M,M,M)
", "ARIMA(1,1,0)(2,1,1)[12]")
final_train
```

```
##                            RMSE     MAE     MAPE    MASE
## snaive                  2.00737 1.95920 0.515780 1.00000
## decompose               0.07694 0.06434 0.016977 0.03284
## Holt-Winters            0.34150 0.27667 0.073054 0.14122
## ETS(M,M,M)              0.34763 0.28170 0.074480 0.14378
## ARIMA(1,1,0)(2,1,1)[12] 0.11979 0.03629 0.009649 0.01852
```

```
final_test <- rbind(a_test_n, a_test_d, a_fc2[2,], a_e4[2,], a_m0[2,])
rownames(final_test) <- c("snaive", "decompose","Holt-Winters", "ETS(M,M,M) "
, "ARIMA(1,1,0)(2,1,1)[12]")
final_test
```

```
##                           RMSE    MAE   MAPE   MASE
## snaive                  6.1429 5.5910 1.4137 2.8537
## decompose               0.6222 0.4955 0.1251 0.2529
## Holt-Winters            0.9336 0.8284 0.2095 0.4228
## ETS(M,M,M)              0.7792 0.6831 0.1727 0.3487
## ARIMA(1,1,0)(2,1,1)[12] 0.5185 0.4078 0.1031 0.2082
```

The selected ARIMA model performs best both on training and testing sets. For this reason, we consider ARIMA(0,1,0)(1,1,0) as a best performing model. This model will be used for generating the forecast.

8. Forecast up to 2020

```
# ARIMA(0,1,0)(1,1,0)
arima_final <- Arima(data[,1], order=c(0,1,0), seasonal=c(1,1,0))
arima_final_f <- forecast(arima_final, h=60)
plot(arima_final_f)
```