Exercise 1

Deborah Kewon

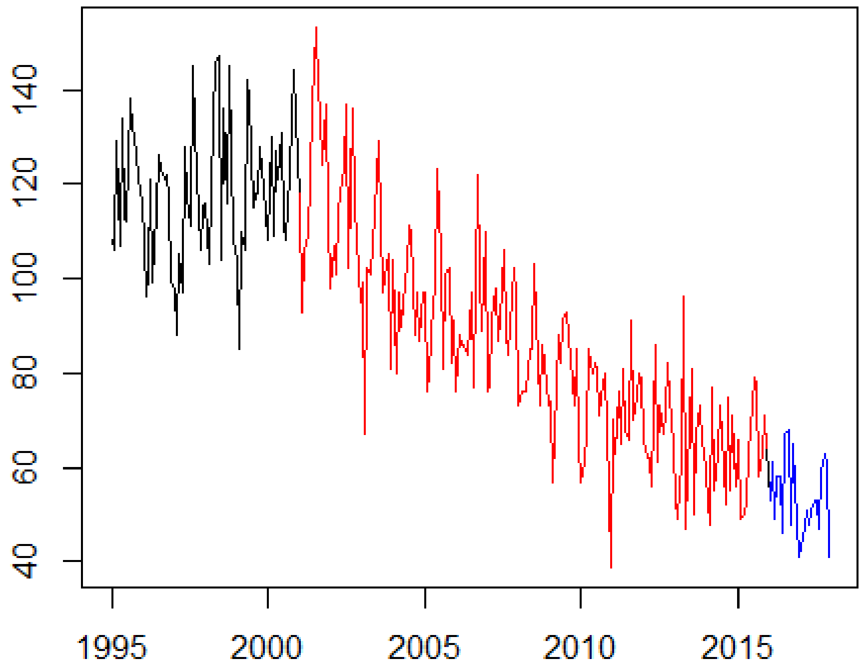
August 10, 2019

The provided dataset contains the monthly number of road fatalities in Belgium from January 1995 to December 2017.The objective is to forecast the number of road fatalities in Belgium (up to December 2020) using various time-series methods such as snaive, stl, ets and arima.

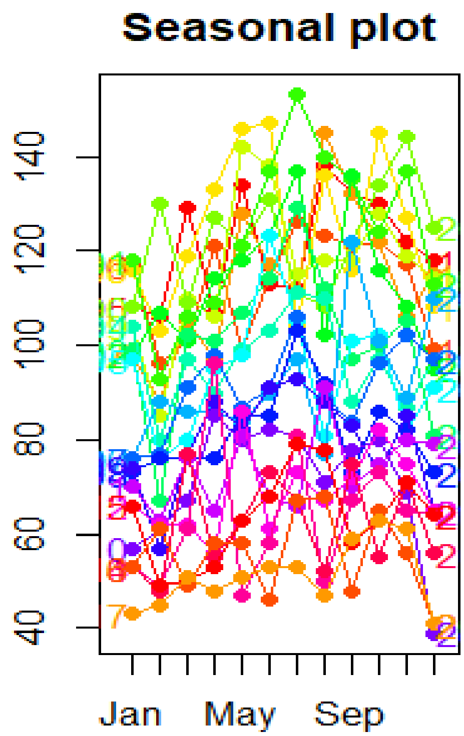
```
if (!require("fpp2")) install.packages("fpp2"); library(fpp2)
if (!require("portes")) install.packages("portes"); library(portes)
if (!require("readxl")) install.packages("readxl"); library(readxl)
if (!require("tseries")) install.packages("tseries"); library(tseries)
if (!require("lmtest")) install.packages("lmtest");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

#importing data
setwd("C:\\Users\\dkewon\\Desktop\\retake\\final")
# read the data
library(readxl)
data <- read_excel("DataSets.xlsx", sheet = "Fatalities_m")
#turning data into time series
rsv <- ts(data[,2], frequency = 12, start = c(1995,1))
# Split the data in training and test set
rsv1 <- window(rsv, start=c(2001,1), end=c(2015,12))
rsv2 <- window(rsv, start=c(2016,1), end=c(2017,12))
# Retrieve the length of the test set
h <- length(rsv2)
```

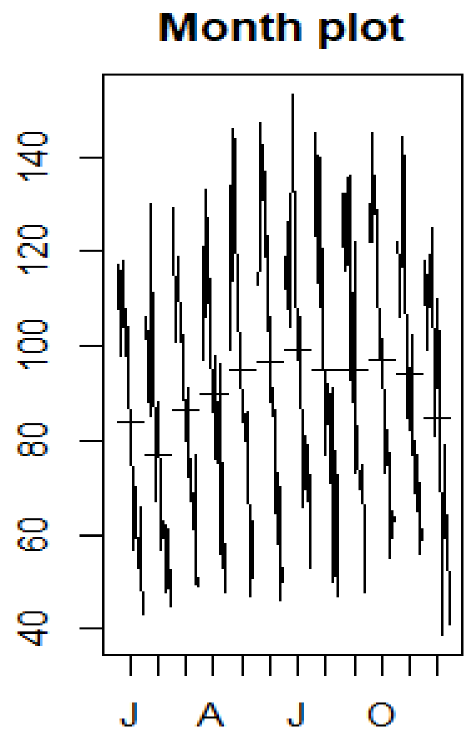
1. Exploring Data

```
# Plot the data
#plotting data to capture seasonal properties
par("mar") #5.1 4.1 4.1 2.1 ## [1] 5.1 4.1 4.1 2.1

## [1] 5.1 4.1 4.1 2.1

par(mar=c(2.5,2.5,2.5,2.5)) #adjusting margin not to get an error message on
large margin
plot(rsv)
lines(rsv1, col="red")
lines(rsv2, col="blue") # moderate trend and high seasonality
```




```r
# looking further into seasonality using season and month plots
par(mfrow=c(1,2))
seasonplot(rsv, year.labels=TRUE, year.labels.left=TRUE,
           main="Seasonal plot",
           ylab="The number of road fatalities",col=rainbow(20), pch=19)
monthplot(rsv, main="Month plot", ylab = "The number of road fatalities",
          xlab="Month", type="l")
```

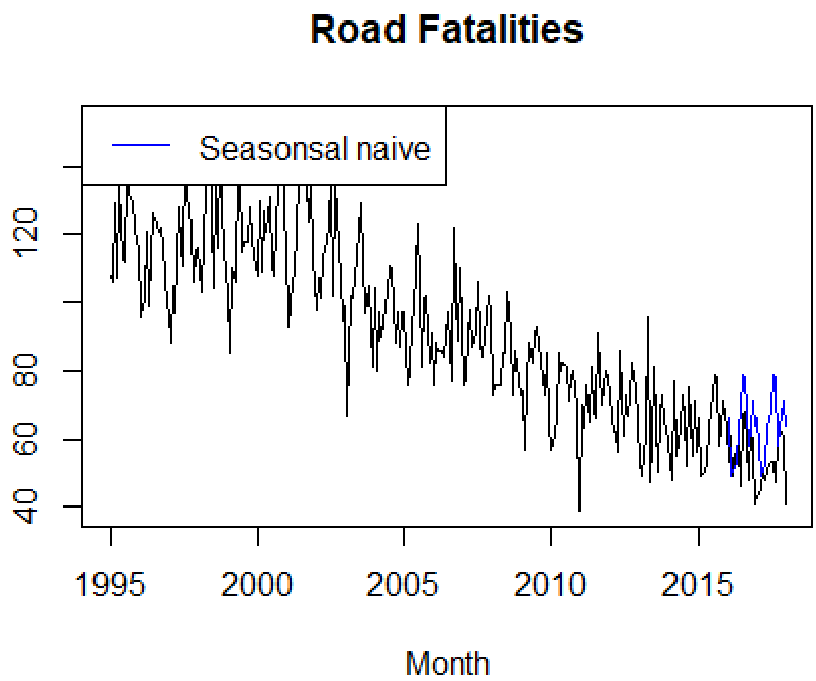




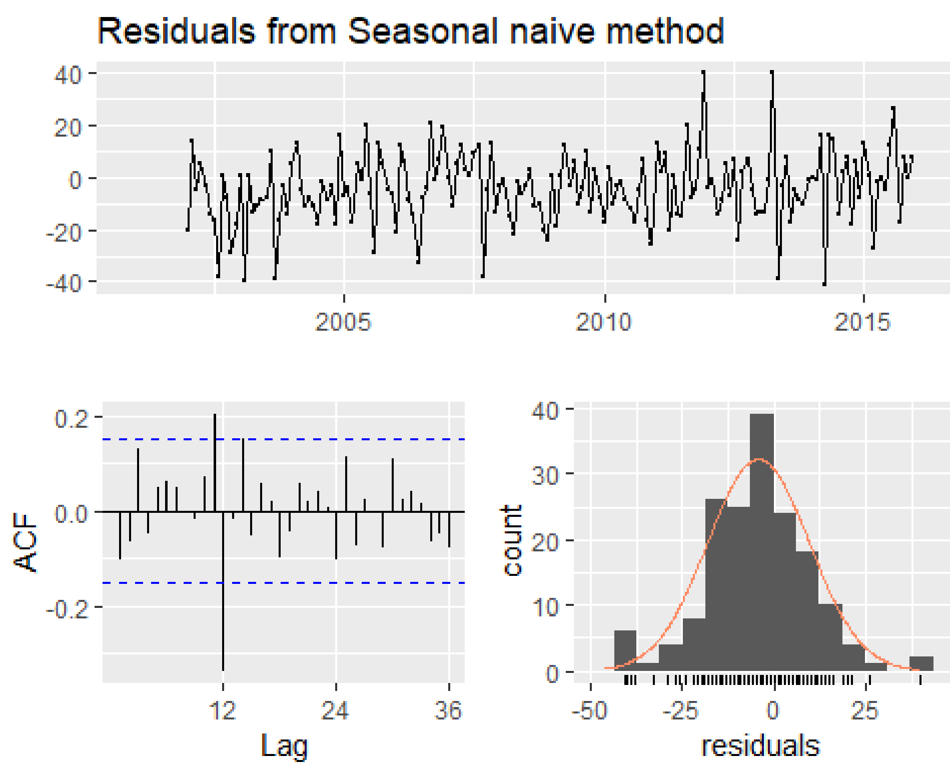
I first split the dataset into trainset (January 2001-December 2015) and testset(January 2016-December 2017) and then looked further into seasonality and trend using season and month plots.Moderate (decreasing) trend and seasonality exist.

2. Seasonal Naive Method

```
n <- snaive(rsv1, h=h) # seasonal naive
a_n <- accuracy(n,rsv2)[,c(2,3,5,6)]
a_train_n <- a_n[1,]
a_train_n

##  RMSE   MAE  MAPE  MASE
## 14.51 11.12 14.51  1.00

a_test_n <- a_n[2,]
a_test_n

##   RMSE    MAE   MAPE   MASE
## 14.659 11.708 23.604  1.053

par(mfrow=c(1,1))
plot(rsv,main="Road Fatalities", 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* = 47, df = 24, p-value = 0.003
##
## Model df: 0.   Total lags used: 24
```

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

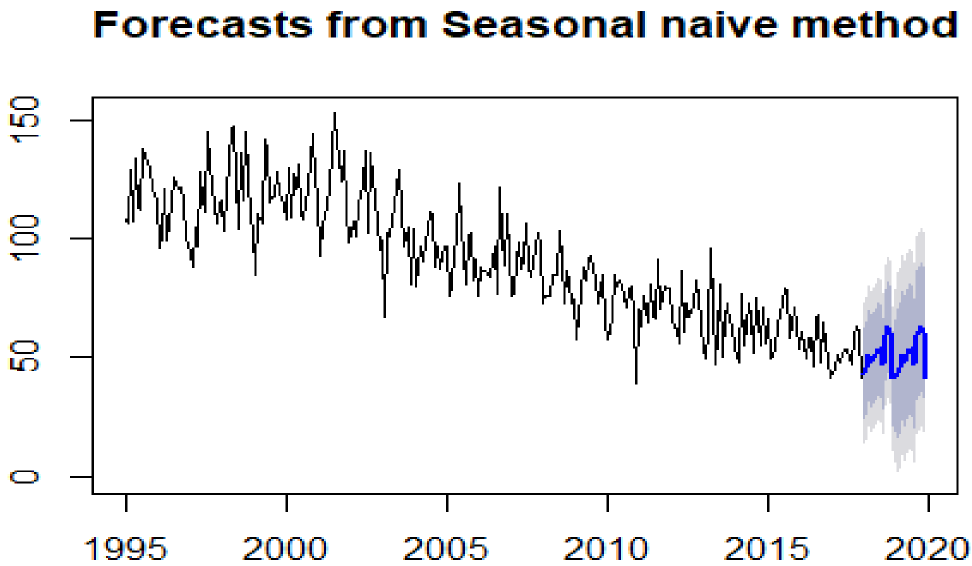
```
##  lags statistic df   p-value
##     1     1.745  1 0.1864924
##     5     6.231  5 0.2844208
##     9     7.475  9 0.5878348
##    13    36.641 13 0.0004715
##    17    42.108 17 0.0006469
##    21    45.065 21 0.0016984
```

The error terms for seasonal naive method are as above. In order to see which method is better in forecasting, we have to compare the error terms of this model with the error terms of other models.

According to the graph above, the number of fatalities obtained from the seasonal naive method are a bit higher than the actual numbers. It is hard to tell how well this method is performing. We may check the quality of residuals.

According to the Auto Correlation Function plot for residuals, there is no white noise, which means there is still information in the residual part. We may find the way to capture this information.

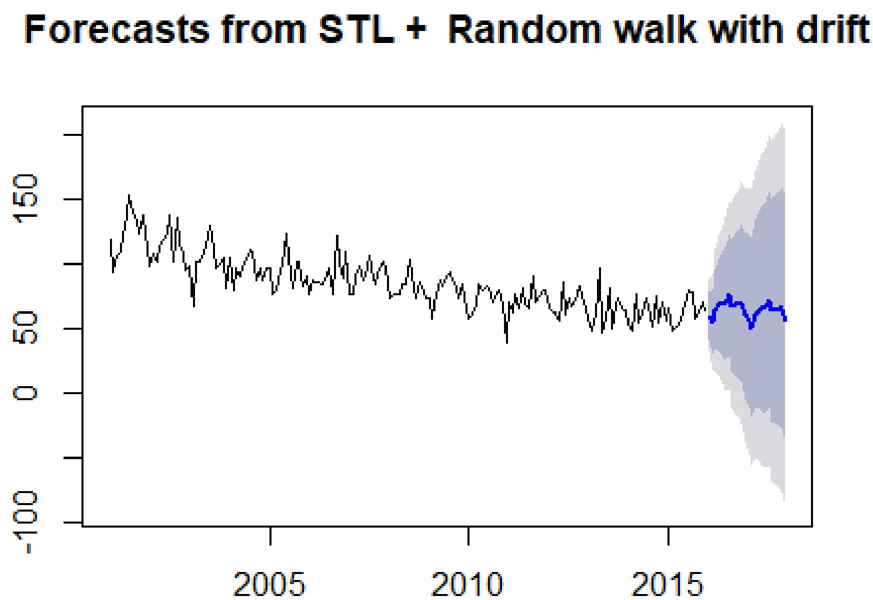
```
par(mfrow=c(1,1))
n_final <- snaive(rsv, h=24)
plot(n_final)
```



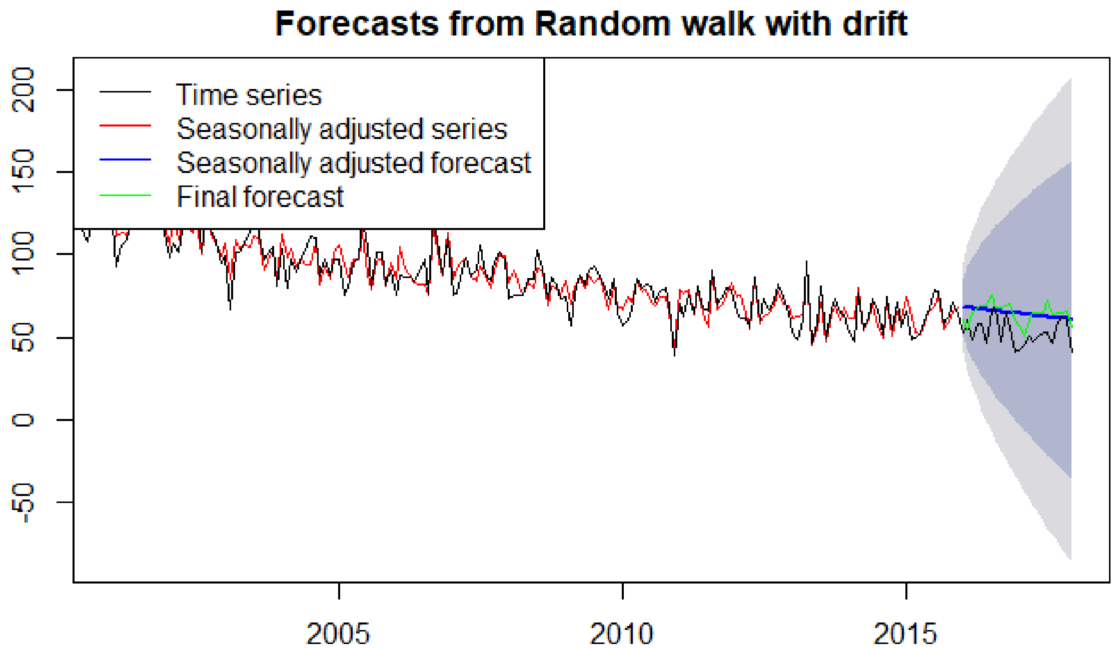
When forecasting on the complete dataset using the seaonal naive, the output looks like this.

3. STL Decomposition

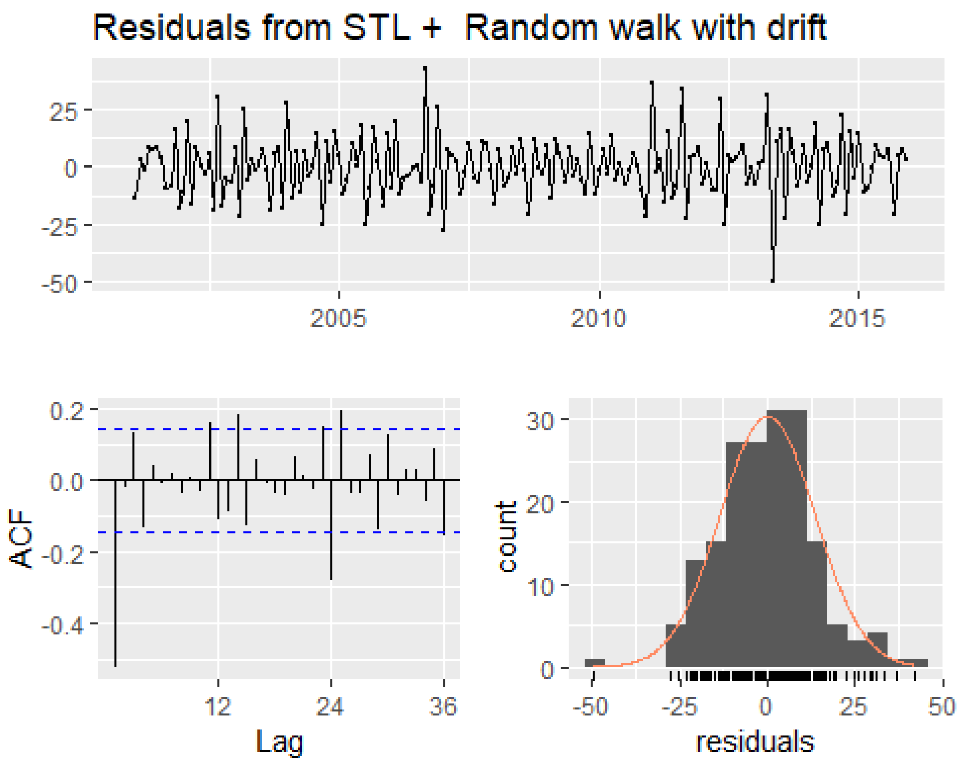
```
d <- stl(rsv1[,1], t.window=15, s.window=13)
rsvadj <- seasadj(d)

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




```r
# In the graph below,
# we plot the various elements that make up the final forecast.
par(mfrow=c(1,1))
plot(rwf(rsvadj, drift=TRUE, h=h), col="red")
lines(rsv, 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 based on a decomposition.
a_d <- accuracy(f_d,rsv2)[,c(2,3,5,6)]
a_train_d <- a_d[1,]
a_train_d

##    RMSE     MAE    MAPE    MASE
## 13.5623 10.5674 13.4008  0.9504

a_test_d <- a_d[2,]
a_test_d

##   RMSE    MAE   MAPE   MASE
## 12.961 11.498 23.122  1.034

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




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk with drift
## Q* = 100, df = 23, p-value = 2e-11
##
## 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     49.88  0 0.000e+00
##     5     56.95  4 1.267e-11
##     9     57.29  8 1.583e-09
##    13     66.35 12 1.534e-09
##    17     76.93 16 5.922e-10
##    21     78.47 20 7.133e-09
```

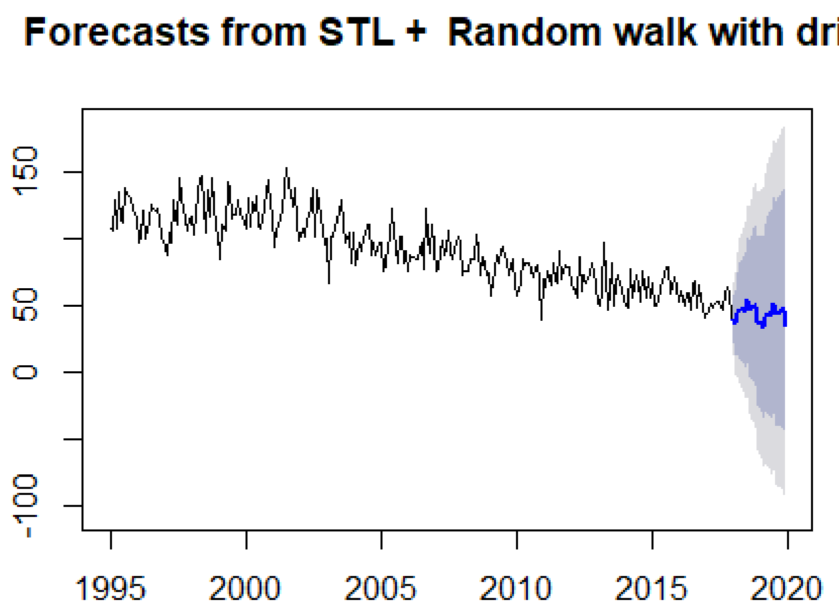
STL (Seasonal decomposition of Time series by Loess) plot decomposes the time series into seasonal, trend and irregular components.The graph above is the result of STL and random walk with draft.

Since the seasonally adjusted forecast does not contain any sesonality components, the line is straight.The final forecast is a bit higher than the actual time series.

The accuracy of this model is better than that of seasonal naive method in terms of RMSE,MAPE and MASE (in the test dataset).

According to ACF, there is no white noise. Further improvements on the forecast model may be needed.

```
d_final <- stl(rsv[,1], t.window=15, s.window=13)
rsvadj <- seasadj(d_final)
f_d_final <- forecast(d_final, method="rwdrift", h=24)
plot(f_d_final)
```

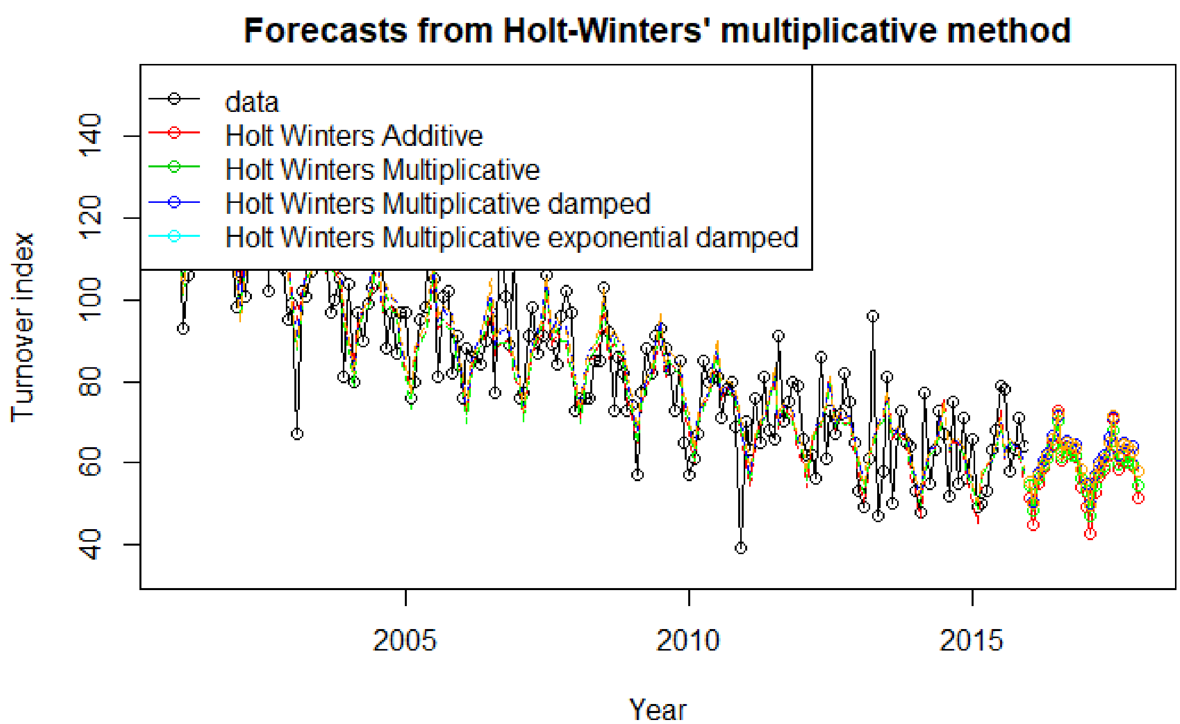


When you apply the STL model on the whole dataset, the output looks like this.

4. Holt_Winter seasonal method

```
fit1 <- hw(rsv1,seasonal="additive",h=h)
fit2 <- hw(rsv1,seasonal="multiplicative",h=h)
fit3 <- hw(rsv1,seasonal="multiplicative", damped=TRUE,h=h)
fit4 <- hw(rsv1,seasonal="multiplicative",exponential=TRUE, damped=TRUE,h=h)

plot(fit2,ylab="Turnover index",
     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 own train set
a_fc1 <- accuracy(fit1)[,c(2,3,5,6)]
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 10.134 7.730 9.607 0.6952
## a_fc2  9.760 7.482 9.373 0.6729
## a_fc3  9.758 7.656 9.721 0.6885
## a_fc4  9.706 7.549 9.625 0.6789
```

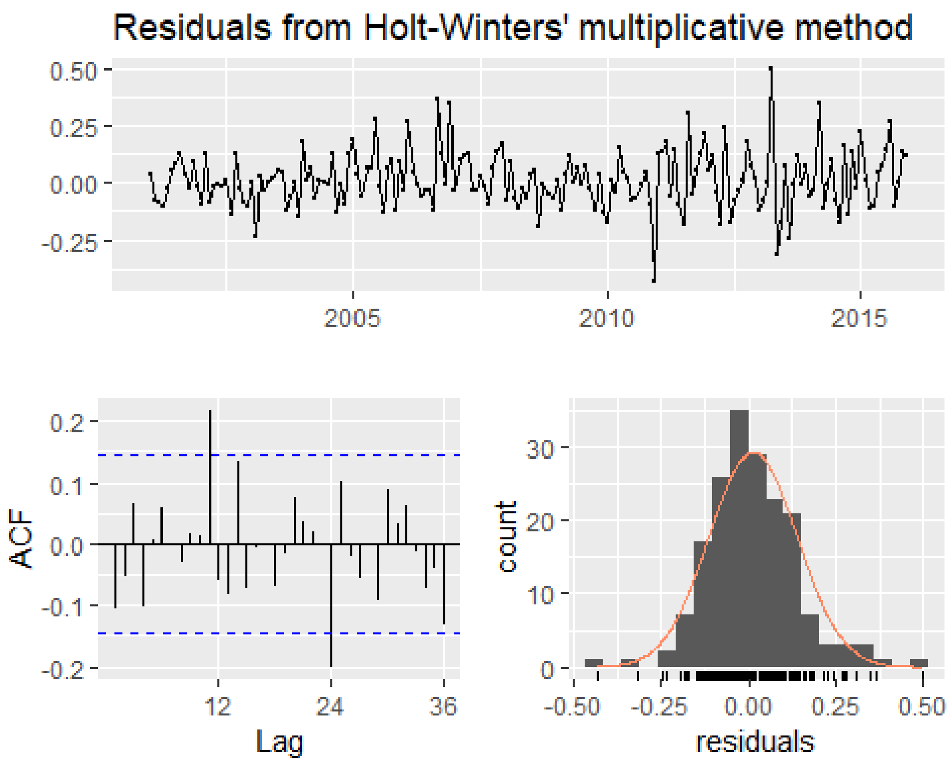
```r
# check acc with test set
a_fc1 <- accuracy(fit1, rsv2)[,c(2,3,5,6)]
a_fc2 <- accuracy(fit2, rsv2)[,c(2,3,5,6)]
a_fc3 <- accuracy(fit3, rsv2)[,c(2,3,5,6)]
a_fc4 <- accuracy(fit4, rsv2)[,c(2,3,5,6)]

acc <- rbind(a_fc1, a_fc2, a_fc3, a_fc4)
acc
```

```
##                RMSE   MAE   MAPE   MASE
## Training set 10.134 7.730  9.607 0.6952
## Test set      9.178 7.416 14.729 0.6669
## Training set  9.760 7.482  9.373 0.6729
## Test set      9.251 7.614 15.405 0.6848
## Training set  9.758 7.656  9.721 0.6885
## Test set     10.969 9.230 18.781 0.8301
## Training set  9.706 7.549  9.625 0.6789
## Test set     10.692 8.887 18.107 0.7993
```

```r
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 1802
## a_fc2 1811
## a_fc3 1810
## a_fc4 1807
```

```r
checkresiduals(fit2)
```

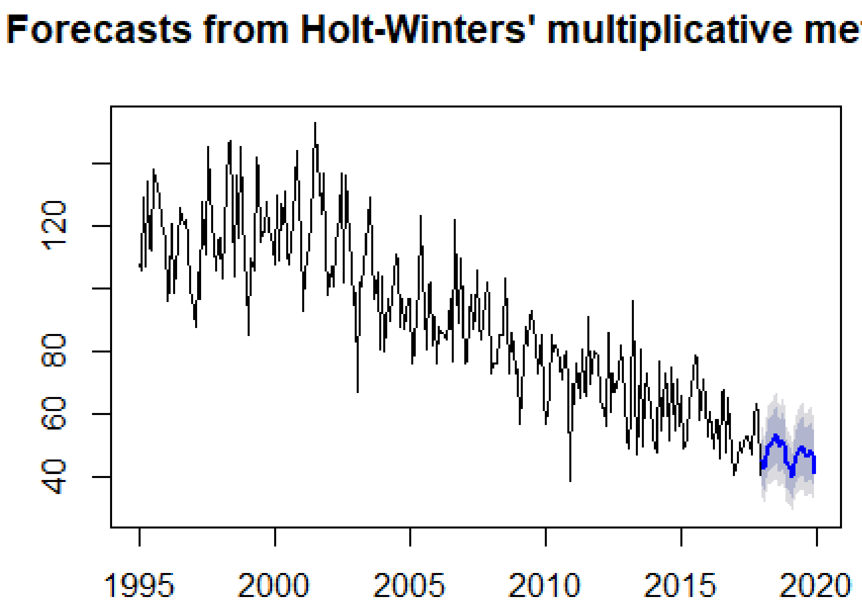



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 33, df = 8, p-value = 6e-05
##
## Model df: 16.   Total lags used: 24

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

##  lags statistic df   p-value
##     1     49.88  0 0.000e+00
##     5     56.95  4 1.267e-11
##     9     57.29  8 1.583e-09
##    13     66.35 12 1.534e-09
##    17     76.93 16 5.922e-10
##    21     78.47 20 7.133e-09
```

In terms of model fit (AIC) and accuracy metrics (RMSE,MAE,MAPE, MASE) for the test set, a_fc1 (Holt Winters' additive method) performs the best.

These residuals still show remaining autocorrelation. The forecast on the complete data set based on this method looks this:

```r
h_final <-  hw(rsv[,1],seasonal="multiplicative")
f_h_final <- forecast(h_final,seasonal="multiplicative", h=24)
plot(f_h_final)
```



5. ETS

```r
#Models without damping (excluding possibly unstable models)
e1 <- ets(rsv1, model="AAA")
e2 <- ets(rsv1, model="MAA")
e3 <- ets(rsv1, model="MAM")
e4 <- ets(rsv1, model="MMM")
#Models with damping (excluding possibly unstable models)
e5 <- ets(rsv1, model="AAA", damped=TRUE)
e6 <- ets(rsv1, model="MAA", damped=TRUE)
e7 <- ets(rsv1, model="MAM", damped=TRUE)
e8 <- ets(rsv1, model="MMM", damped=TRUE)

#AICc as a model fit criteria and Error terms for accuracy criteria


m <- c("AAA", "MAA", "MAM", "MMM")
result <- matrix(data=NA, nrow=4, ncol=9)
for (i in 1:4){
  model <- ets(rsv1, model=m[i], damped=FALSE)
  f <- forecast(model, h=length(rsv2))
  a <- accuracy(f, rsv2)
```

```r
  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
## 1806 1823 1814 1807
```

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

```
##       RMSE   MAE  MAPE   MASE
## AAA 10.134 7.730 9.607 0.6952
## MAA 10.160 7.720 9.590 0.6943
## MAM  9.881 7.579 9.414 0.6816
## MMM  9.726 7.544 9.472 0.6785
```

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

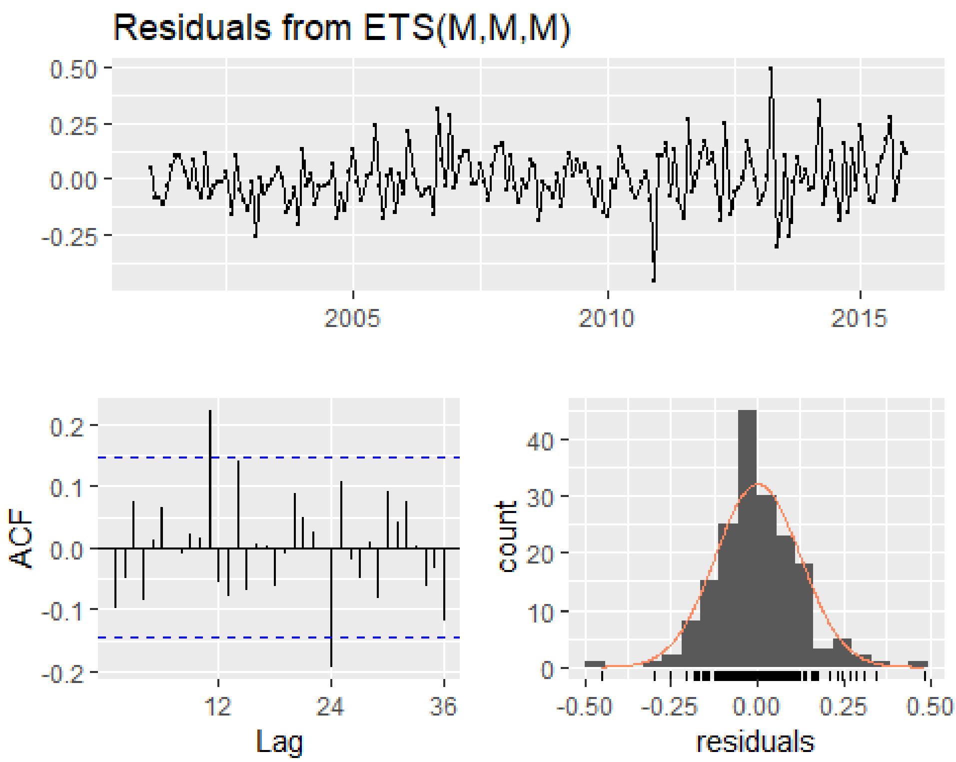
```
##      RMSE   MAE  MAPE   MASE
## AAA 9.178 7.416 14.73 0.6669
## MAA 9.496 7.688 15.32 0.6914
## MAM 8.885 7.291 14.81 0.6557
## MMM 7.802 6.150 12.31 0.5531
```

```r
# same procedure for the damped models


m <- c("AAA", "MAA", "MAM", "MMM")
result <- matrix(data=NA, nrow=4, ncol=9)
for (i in 1:4){
  model <- ets(rsv1, model=m[i], damped=TRUE)
  f <- forecast(model, h=length(rsv2))
  a <- accuracy(f, rsv2)
  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
## 1806 1819 1816 1810

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

##        RMSE   MAE   MAPE   MASE
## AAdA 10.051 7.781  9.865 0.6998
## MAdA 10.135 7.879 10.049 0.7086
## MAdM  9.909 7.813  9.927 0.7027
## MMdM  9.751 7.612  9.693 0.6846

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

##       RMSE   MAE  MAPE   MASE
## AAdA 10.79 8.802 17.60 0.7916
## MAdA 11.65 9.732 19.47 0.8752
## MAdM 11.31 9.669 19.60 0.8696
## MMdM 10.54 8.787 17.91 0.7902

# The damped models have higher error terms than non-damped ones in all cases
. Therefore, we will use non-damped models


summary(e4)

## ETS(M,M,M)
##
## Call:
##  ets(y = rsv1, model = "MMM")
##
##   Smoothing parameters:
##     alpha = 0.0466
##     beta  = 1e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 126.8953
##     b = 0.9958
##     s = 0.9761 1.041 1.056 1.068 1.038 1.14
##            1.057 1.012 0.9911 0.9229 0.8033 0.8926
```

```
##
##   sigma:  0.1283
##
##   AIC AICc  BIC
## 1803 1807 1858
##
## Training set error measures:
##                     ME  RMSE   MAE    MPE  MAPE   MASE     ACF1
## Training set -0.2639 9.726 7.544 -1.461 9.472 0.6785 -0.05719
```

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

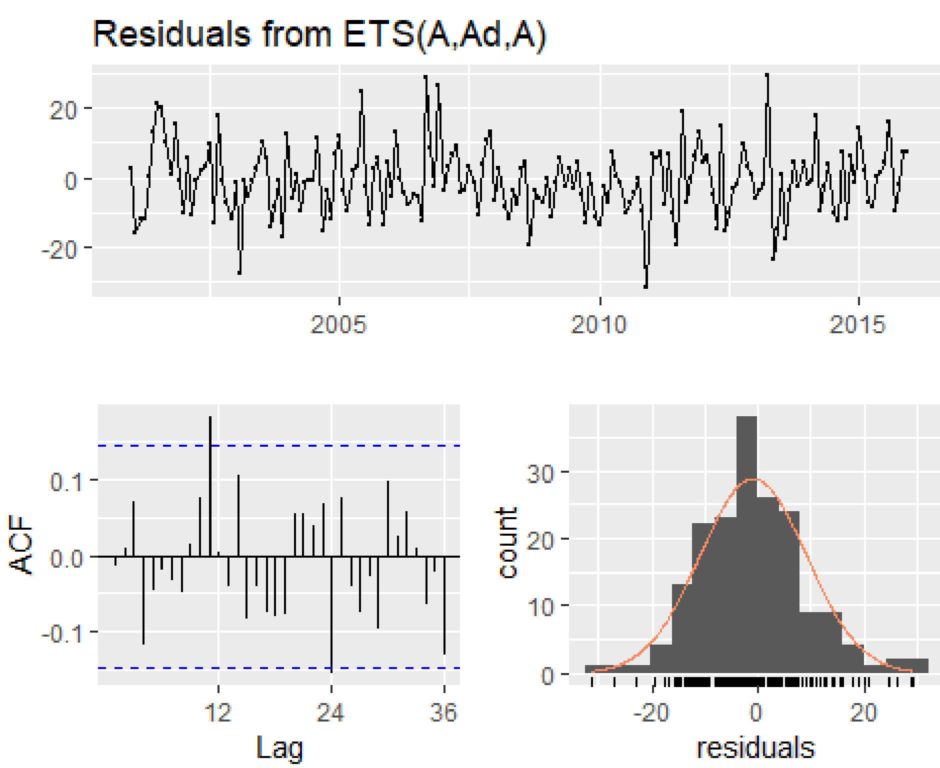



```
##
##   Ljung-Box test
##
## data:  Residuals from ETS(M,M,M)
## Q* = 33, df = 8, p-value = 8e-05
##
## Model df: 16.   Total lags used: 24
```

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

```
##   lags statistic df   p-value
##     16     21.76  0 0.000e+00
```

```
##    20      24.11  4 7.594e-05
##    24      32.51  8 7.530e-05

# For these residuals, we do reject the null hypothesis of white noise.
# We compare the results with those of the automated ETS procedure.

auto_ets <- ets(rsv1)
auto_ets$method

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

f <- forecast(auto_ets, h=length(rsv2))
accuracy(f, rsv2)[,c(2,6)]

##                 RMSE   MASE
## Training set 10.05 0.6998
## Test set     10.79 0.7916

checkresiduals(auto_ets)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,A)
## Q* = 28, df = 7, p-value = 2e-04
##
## Model df: 17.   Total lags used: 24
```

```r
# The MAM model from auto ets is not a good performing model in terms of accu
racy
```

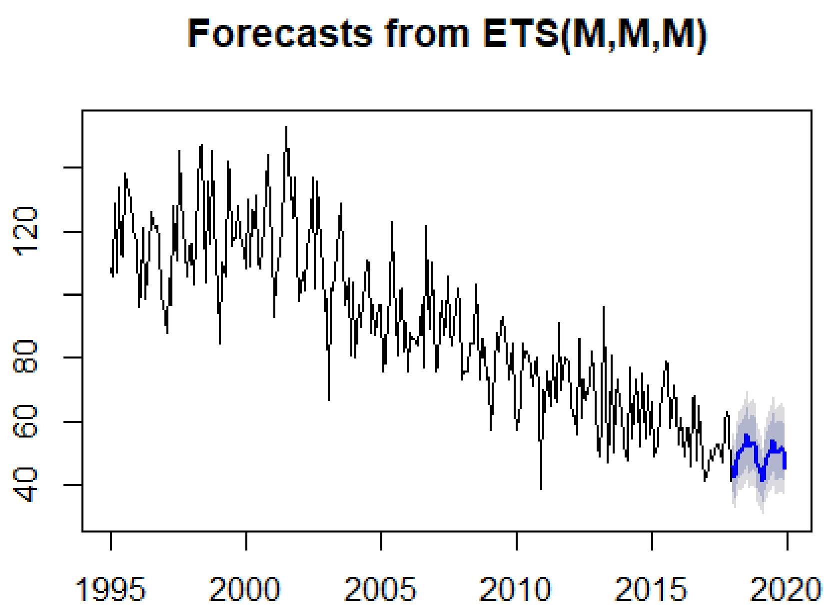
The ETS model without damping is performed.MMM performs the best for the testset in terms of RMSE,MAE,MAPE and MASE even though there is no white noise.

Damped models have higher error terms in all cases. As a result, I select non-damped MMM model based on the accuracy metrics.

MAM model from auto ETS does not perform well in terms of accuracy metrics. Therefore,as I mentioned earlier, my final choice is the MMM model (e4) despite that residuals of this model do not perform well.

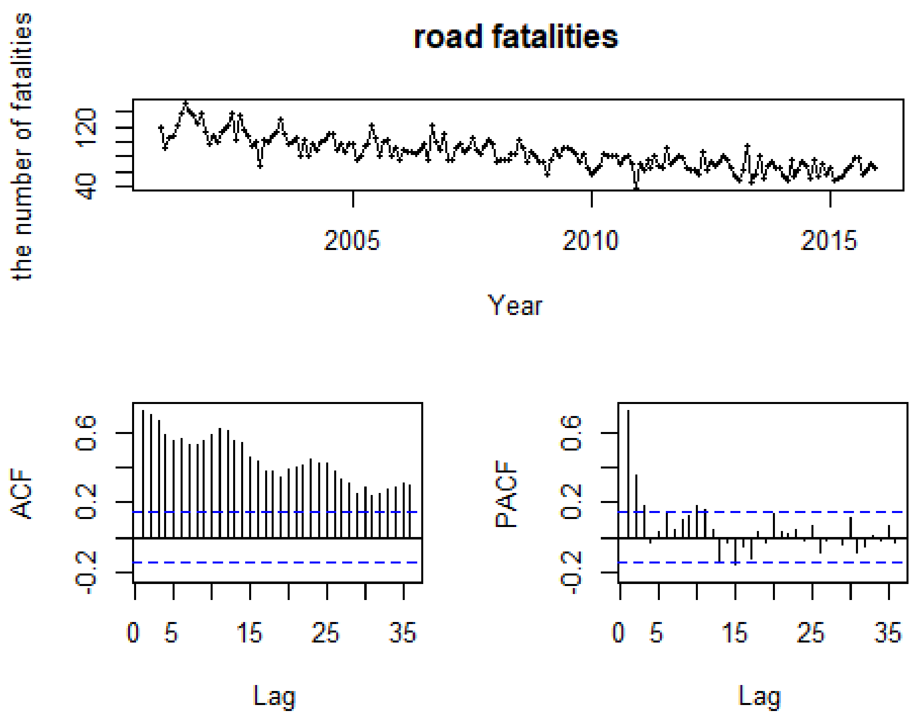
We will apply this model to the complete data set.

```r
e4 <- ets(rsv1, model="MMM",damped = FALSE)
f_e4 <- forecast(e4, h=length(rsv2))
a_e4 <- accuracy(f_e4,rsv2)[,c(2,3,5,6)]

e_final <- ets(rsv[,1], model = "MMM",damped = FALSE)
e_final_f <- forecast(e_final, h=24)
plot(e_final_f)
```

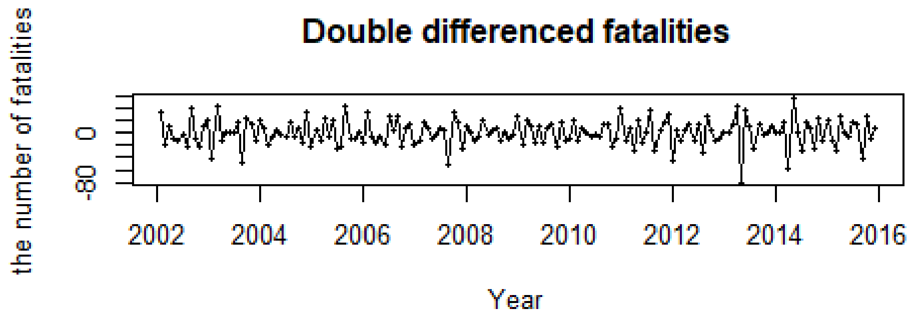


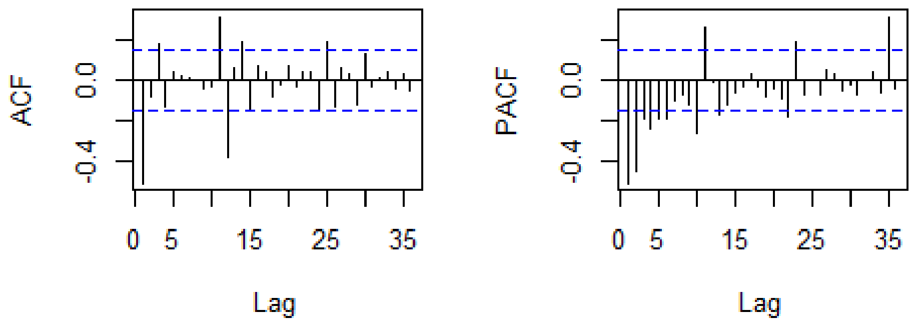
6. ARIMA

```
tsdisplay(rsv1, main="road fatalities", ylab="the number of fatalities", xlab
="Year") #ACF shows  'nonstationary', which is caused by seasonality.
```




```
ndiffs(rsv1)

## [1] 1

nsdiffs(diff(rsv1))

## [1] 0

tsdisplay(diff(diff(rsv1,12)), main="Double differenced fatalities",
          ylab="the number of fatalities", xlab="Year")
```



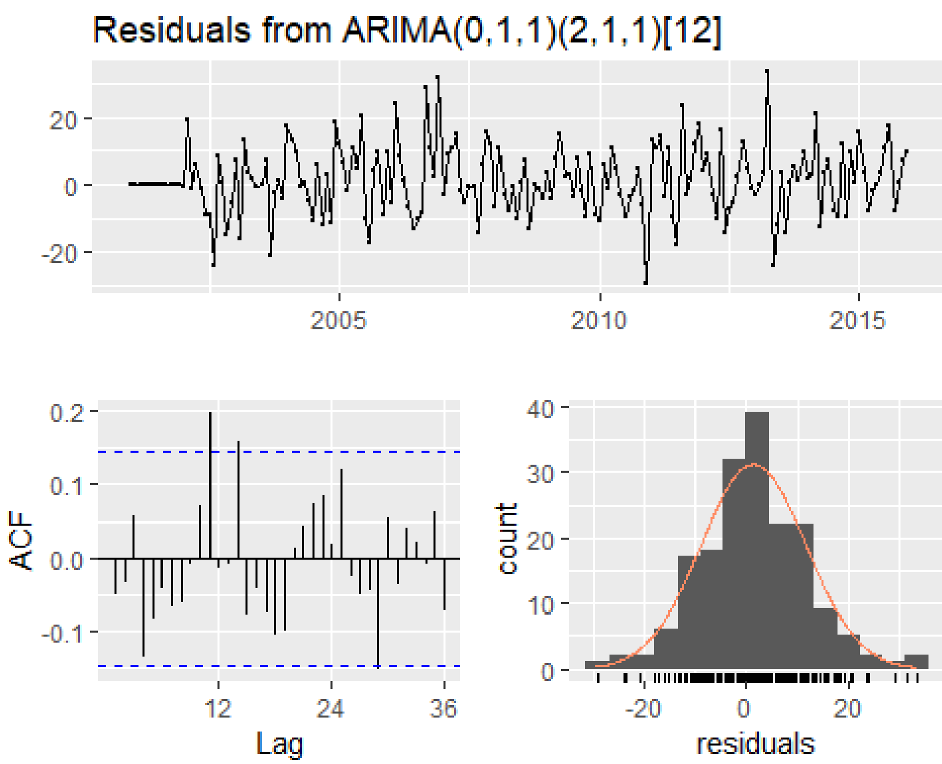



```
#As this dataset includes seasonal components, I used ndiffs to estimate the
number of differences required to make the given time series stationary-1 dif
ference.


# define function
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)
}

# for loop to save it as matrix
mat <- matrix(NA,nrow=54, ncol=5)
modelnames <- vector(mode="character", length=54)
```

```r
line <- 0
for (i in 2:4){
  for (j in 0:2){
    for (k in 0:1){
      for (l in 0:2){
        line <- line+1
        mat[line,] <- getinfo(rsv,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 a 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] "1884"                          "0.6969"
## [3] "0.7131"                        "11.04"
## [5] "10.633"                        "ARIMA(4,1,2)(0,1,1)[12]"

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

## [1] "1887"                          "0.6883"
## [3] "0.6289"                        "10.92"
## [5] " 9.704"                        "ARIMA(4,1,1)(1,1,1)[12]"

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

## [1] "1888"                          "0.7146"
## [3] "0.5904"                        "11.22"
## [5] " 9.146"                        "ARIMA(3,1,2)(0,1,1)[12]"

# proceed with the auto.arima procedure
m0 <- auto.arima(rsv1, stepwise = FALSE, approximation = FALSE, d=1, D=1)
m0

## Series: rsv1
## ARIMA(0,1,1)(2,1,1)[12]
##
```

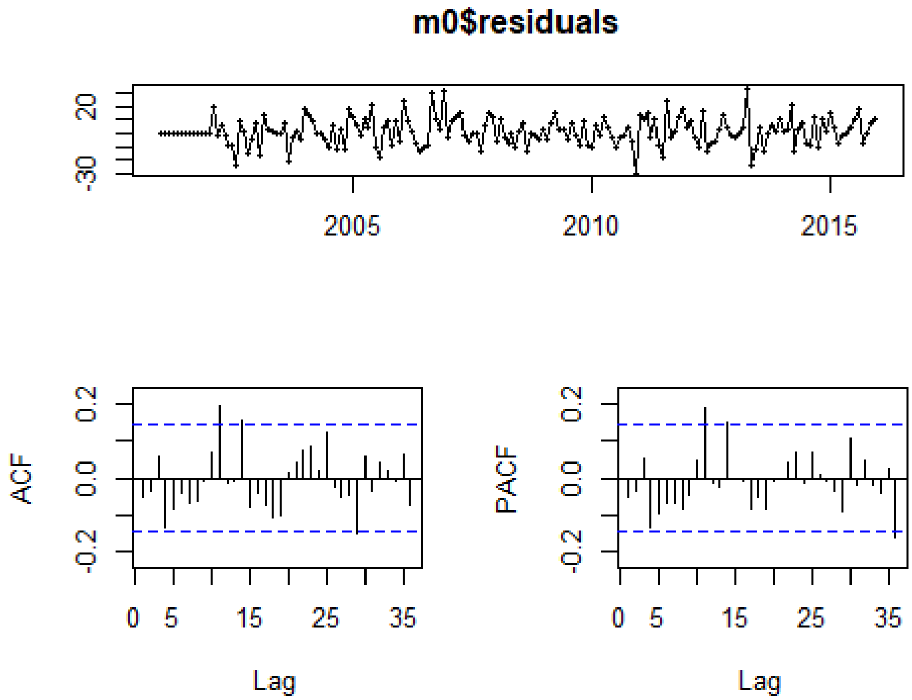
```
## Coefficients:
##           ma1    sar1     sar2     sma1
##        -0.911   0.038   -0.145   -0.856
## s.e.    0.034   0.101    0.095    0.106
##
## sigma^2 estimated as 120:  log likelihood=-644.8
## AIC=1300   AICc=1300   BIC=1315
```

```
checkresiduals(m0)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(2,1,1)[12]
## Q* = 31, df = 20, p-value = 0.05
##
## 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     4.706  0 0.00000
##     8     7.843  4 0.09749
##    12    16.411  8 0.03686
##    16    22.895 12 0.02863
##    20    28.292 16 0.02917
##    24    31.392 20 0.05022

f0 <- forecast(m0, h=h)
accuracy(f0,rsv2)[,c(2,3,5,6)]

##                RMSE   MAE  MAPE   MASE
## Training set 10.417 7.852 10.01 0.7062
## Test set      7.561 6.036 11.94 0.5428
```

Based on these results, we select 3 models 1. m0: ARIMA(0,1,1)(2,1,1)[12] is the model selected by auto.arima. 2. m1: ARIMA(4,1,2)(0,1,1)[12] shows the best AICc. 3. m2: ARIMA(3,1,2)(0,1,1)[12] It has the lowest error terms for the test set.We now study these selected models in more detail.

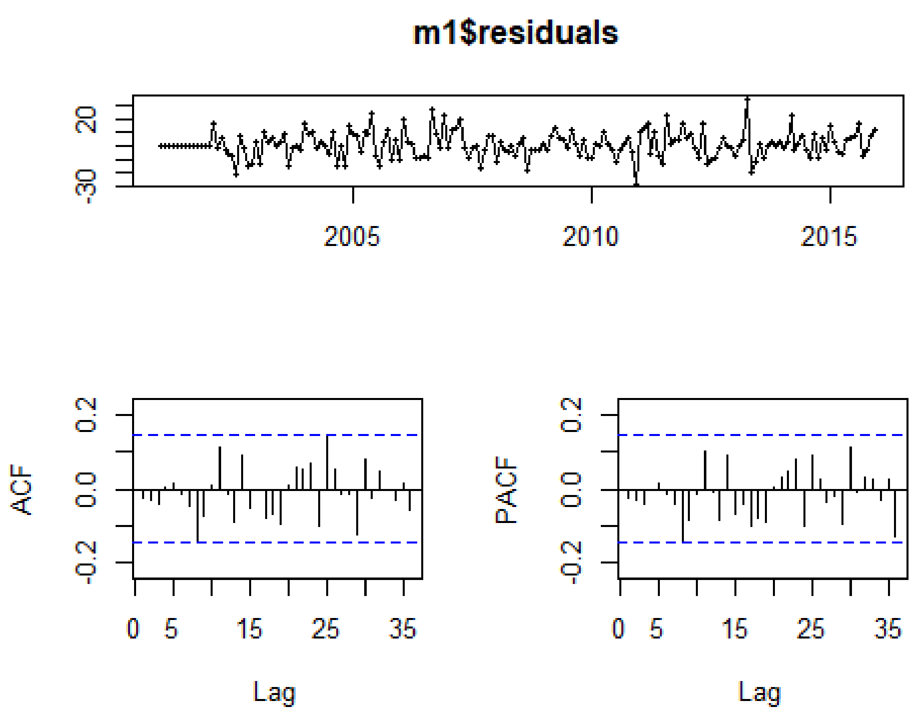
```
m1 <- Arima(rsv1, order=c(4,1,2), seasonal=c(0,1,1))
coeftest(m1)

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

```
##       Estimate Std. Error z value Pr(>|z|)
## ar1   0.76302    0.12090    6.31  2.8e-10 ***
## ar2  -0.03226    0.09709   -0.33   0.7396
## ar3   0.00949    0.09681    0.10   0.9219
## ar4  -0.25377    0.09329   -2.72   0.0065 **
## ma1  -1.70707    0.09193  -18.57  < 2e-16 ***
## ma2   0.78920    0.09953    7.93  2.2e-15 ***
## sma1 -0.99991    0.17907   -5.58  2.4e-08 ***
## ---
## 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
##     7    0.9716  0  0.0000
##    11    8.3582  4  0.0793
##    15   12.2122  8  0.1420
##    19   16.1509 12  0.1844
##    23   18.4394 16  0.2988
```

```
tsdisplay(m1$residuals)
```



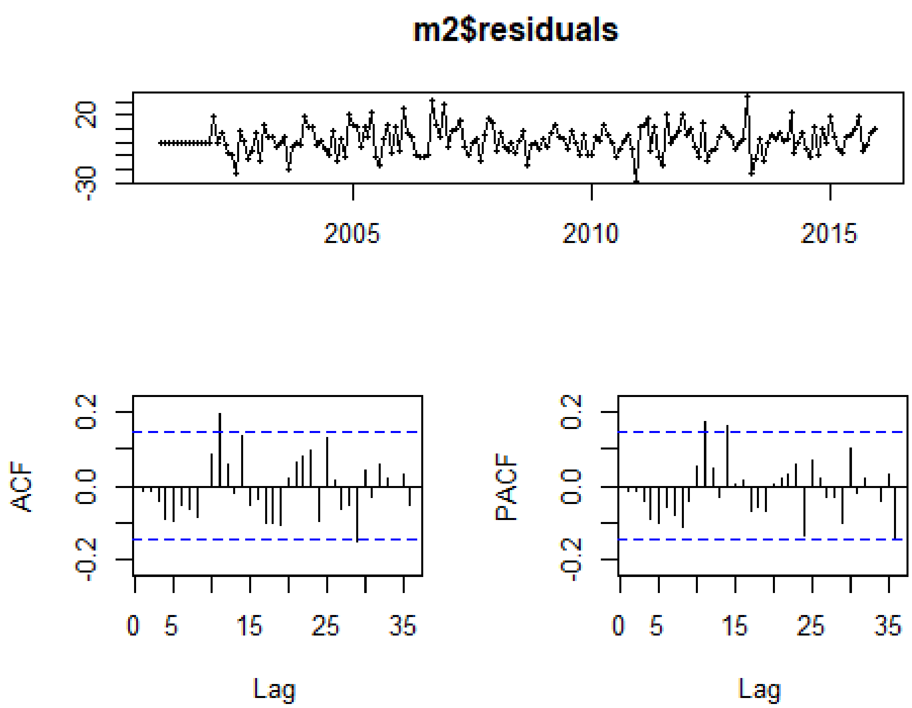

```
f1 <- forecast(m1, h=h)

m2 <- Arima(rsv1, order=c(3,1,2), seasonal=c(0,1,1))
coeftest(m2)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error z value Pr(>|z|)
## ar1   -0.6078     0.5386   -1.13     0.26
## ar2   -0.0327     0.1173   -0.28     0.78
## ar3    0.0810     0.0981    0.82     0.41
## ma1   -0.3420     0.5371   -0.64     0.52
## ma2   -0.5142     0.4746   -1.08     0.28
## sma1  -0.8741     0.0906   -9.65   <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
##     6     3.976  0 0.00000
##    10     7.450  4 0.11394
##    14    19.365  8 0.01302
##    18    24.292 12 0.01856
##    22    28.898 16 0.02463
```

```
tsdisplay(m2$residuals)
```




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

#relevant accuracy measures.
```

```r
a_m0 <- accuracy(f0,rsv2)[,c(2,3,5,6)]
a_m1 <- accuracy(f1,rsv2)[,c(2,3,5,6)]
a_m2 <- accuracy(f2,rsv2)[,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 10.417 7.852 10.011 0.7062
## a_m1  9.862 7.400  9.407 0.6655
## a_m2 10.470 8.036 10.210 0.7228

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 7.561 6.036 11.94 0.5428
## a_m1 9.620 7.564 15.16 0.6803
## a_m2 8.286 6.664 13.21 0.5993
```

We observe that the requirements of white noise residuals are fulfilled in m1. However, a_m0 from auto arima performs the best in terms of accuracy metrics. Therefore I select m0: ARIMA(0,1,1)(2,1,1)[12] as a final model.

7. Model Comparison & Sample Forecast up to December 2020

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

```r
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(0,1,1)(2,1,1)
[12]")
final_train

##                            RMSE    MAE   MAPE   MASE
## snaive                   14.508 11.119 14.506 1.0000
## decompose                13.562 10.567 13.401 0.9504
## Holt-Winters              9.760  7.482  9.373 0.6729
## ETS(M,M,M)                9.726  7.544  9.472 0.6785
## ARIMA(0,1,1)(2,1,1)[12] 10.417  7.852 10.011 0.7062

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(0,1,1)(2,1,1)[12]")
final_test
```

```
##                              RMSE    MAE  MAPE   MASE
## snaive                     14.659 11.708 23.60 1.0530
## decompose                  12.961 11.498 23.12 1.0341
## Holt-Winters                9.251  7.614 15.41 0.6848
## ETS(M,M,M)                  7.802  6.150 12.31 0.5531
## ARIMA(0,1,1)(2,1,1)[12]     7.561  6.036 11.94 0.5428
```

We observe that the Holt-Winters performs best on the training set in terms of MAE,MAPE and MASE. However, on the test set, the best forecast accuracy is the ARIMA(0,1,1)(2,1,1)[12].

The residual diagnostics were not satisfactory for both models, we reject the null hypothesis of white noise residuals

Therefore, I chose m0:ARIMA(0,1,1)(2,1,1)[12] as a final model for generating the forecasts up to 2020.

We select ARIMA(0,1,1)(2,1,1)[12] as the final model for generating the forecasts up to December 2020 (See below)

```
e_final_f <- forecast(m0, h=60)
plot(e_final_f)
```